Quiz Submissions - Quizzterm # 1 Objective

Top of Form

**Matthew Zaldana (username: matthew.zaldana.050242D559A8K8)**

**Attempt 1**

Written: May 27, 2021 1:11 PM - May 27, 2021 1:31 PM

**Submission View**

Your quiz has been submitted successfully.

|  |  |  |
| --- | --- | --- |
| **Question 1** |  | 1 / 1 point |

Suppose that we want to manage soccer scores in a table.  We can say the following about an attribute value like "home: 5, away: 3" (check all that apply):

|  |  |  |
| --- | --- | --- |
|  | This value actually represents **two** individual concepts: the home team score and the away team score.  As such, it should be separated into two attributes. | |
|  | Is difficult to validate.  Perhaps someone forgot to enter the word home, or left out the colon, or the numerical value of the score was not an integer. | |
|  | Is more efficient because it stores more than one piece of information in just one column.  That makes this a good approach. | |
|  | It is difficult to search such values.  For instance, to find all of the games in which the winner won by more than a 3 point spread would be very difficult. | |
| **Question 2** |  | 0 / 1 point | |

While the alter table statements gives us the **ability** to change a table to add a primary key, it is always best to define the primary key when we define the table so that there is never any opportunity to insert redundant rows into the table.

|  |  |  |  |
| --- | --- | --- | --- |
| Correct Answer |  | True | |
|  |  | False | |
| **Question 3** | |  | 0 / 1 point | |

In this course, you **must** include the column names in your insert statements.

|  |  |  |  |
| --- | --- | --- | --- |
| Correct Answer |  | True | |
|  |  | False | |
| **Question 4** | |  | 1 / 1 point | |

Match the following terms to their respective definition.  Each term and each definition can only be once.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | |  | \_\_6\_\_ |  | is used to build and manipulate relational databases. It is a declarative, rather than a procedural, programming language. In a declarative language, statements express/describe/declare what is to be done, rather than provide the specifics of how something is to be done as is the case in procedural languages. Internally, database systems translate SQL to relational algebra. There is a standard for SQL, but database vendors vary in how closely they implement it. This website emphasises the standard and thus examples presented can be executed on all relational database systems with some minor (if necessary at all) changes. | |  | \_\_4\_\_ |  | is a formal language used to symbolically manipulate elements of the relational model. As the name suggests it is an algebra like the familiar algebra that operates on numbers, but this one operates on relations that have the data in our relational database. | |  | \_\_2\_\_ |  | model is used in many database development systems. There are many different graphic standards that can represent this model. Some of the most modern of these look very similar to the UML class diagram, but may also include elements of the relational model. | |  | \_\_5\_\_ |  | is an informal set of terms for relational model objects. These are the terms used most often by database developers. | |  | \_\_1\_\_ |  | was designed for software engineering of large systems using object-oriented (OO) programming languages.  This language is a very large language; we will use only a small portion of it here, to model those portions of an enterprise that will be represented in the database.  It is our tool for communicating with the client in terms that are used in the enterprise and for communicating with a database developer whose job it is to implement the database modeled. | |  | \_\_3\_\_ |  | is the formal model of a database that was developed for IBM in the early 1970s by Dr. E.F. Codd. It is largely based on set theory, which makes it both powerful and easy to implement in computers. All modern relational databases are based on this model. We will use it to represent information that does not (and should not) appear in the UML model but is needed for us to build functioning databases. | |  | |  |  | | --- | --- | | **1**. | Unified Modeling Language (UML) | | **2**. | Entity-Relationship | | **3**. | Relational Model (RM) | | **4**. | Relational Algebra (RA) | | **5**. | table model | | **6**. | Structured Query Language (SQL) | | |
| **Question 5** | |  | 1 / 1 point | |

The Universal Modeling Language (UML) class diagram contains the following (check all that apply):

|  |  |  |
| --- | --- | --- |
|  | The class name. | |
|  | Storage estimates for the table that will implement the class. | |
|  | Primary key of the class. | |
|  | Descriptive attributes of the class. | |
| **Question 6** |  | 1 / 1 point | |

The following can be said about *sets* (check all that apply):

|  |  |  |
| --- | --- | --- |
|  | The elements in a set are not ordered. | |
|  | A set can**not** contain duplicate elements. | |
|  | Each set has membership rules that tell us whether a given element belongs to that set or not. | |
|  | Any set my be manipulated with the usual set operators, such as union, intersection and minus. | |
| **Question 7** |  | 1 / 1 point | |

The SQL select statement has the following clauses in it (check all that apply):

|  |  |  |
| --- | --- | --- |
|  | select | |
|  | from | |
|  | exclude (followed by a Boolean expression to indicate rows that we want filtered **out**) | |
|  | order by | |
|  | where | |
| **Question 8** |  | 5 / 5 points | |

Specify the definition that goes with each term.

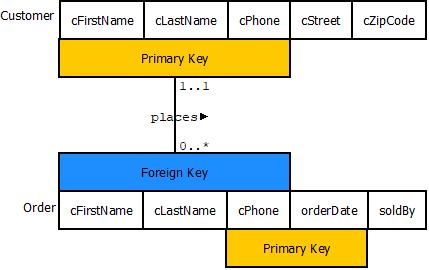
1. primary key
2. surrogate key
3. candidate key
4. external key
5. super key

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | \_\_5\_\_ |  | Any set of attributes whose values, taken together, uniquely identify each row of a table. | |
|  | \_\_1\_\_ |  | The set of one or more attributes that the designer selects to uniquely identify the table and which will migrate into any child tables. | |
|  | \_\_4\_\_ |  | A *surrogate* or *substitute* key that has been defined by an external organization. May be treated as a *descriptive attribute* in your model. | |
|  | \_\_2\_\_ |  | An artificial, meaningless, primary key made up by the database designer under certain limited conditions. | |
|  | \_\_3\_\_ |  | A minimal super key, candidate to become the primary key. | |
| **Question 9** | | |  | 0 / 1 point | |

The term *class* as we use it in this course has much in common with the use of that term in your Java or C++ coding, including (check all that apply):

|  |  |
| --- | --- |
| Correct AnswerIncorrect Response | A given attribute in a class, like model\_year for a Car class, means the same thing for each of the objects (rows) within that class. |
| Correct Answer | A class serves as a template, or cookie cutter for constructing instances of that class. |
| Correct Answer | A class has a set of criteria that one can apply conceptually to determine whether a given instance belongs to the class or not. |
| Correct AnswerIncorrect Response | What **differs** from one instance of a class to the next is the **state** of those instances. |
| **Question 10** | |  | 0 / 1 point |

Consider the following relation scheme diagram:



Check all of the following statements that apply:

|  |  |
| --- | --- |
| Incorrect Response | The chances of getting two customers with the same phone number submitting an order on the same date are astronomically low, so the primary key of Order is fine. |
| Incorrect Response | In most cases, the customer's phone number is unique.  No one shares a phone number with anyone else.  So there is no problem just putting the cphone from Customer into the primary key of Order. |
| Correct AnswerIncorrect Response | We must be able to tell which customer placed the order.  If cPhone is unique, then the primary key for Customer is not minimal.  If cPhone does not uniquely identify the Customer, then Order could have duplicate rows. |
| Correct AnswerIncorrect Response | The entirety of the primary key of Customer either migrates into the primary key of Order or it doesn't. |
| **Question 11** | |  | 0 / 1 point |

If you can**not** find **any** set of attributes in a relation scheme that uniquely identifies that relation scheme, it's better to just not give it a primary key at all.

|  |  |  |
| --- | --- | --- |
| Incorrect Response |  | True |
| Correct Answer |  | False |
| **Question 12** | | |  | 1 / 1 point |

It is always best to create the UML model first, since that is conceptual in nature, then go to the relation scheme diagram since that is closer to the physical implementation because it captures the keys in the relation scheme, and **then** do the physical create table statement.

|  |  |  |
| --- | --- | --- |
|  |  | True |
|  |  | False |

Bottom of Form