**SQL Assignment 7.6 – INTRO Assignment Final**

1. The list of data types (CHAR(n), TEXT(n), VARCHAR(n)) can store alphanumeric data that is either fixed at n symbols or up to n symbols in length. These data types are used for storing text or string values. The "CHAR" type represents a fixed-length string of length n, while "TEXT" represents a variable-length string with a maximum length of n. "VARCHAR" also represents a variable-length string with a maximum length of n. Arithmetic operations cannot be performed directly on these data types.
2. The list of data types (REAL, FLOAT, NUMBER, NUMERIC, DECIMAL) can store numbers with decimal places. These data types are used for representing floating-point or decimal values. They can hold both positive and negative numbers, as well as fractions or decimal values. Arithmetic operations and numeric comparisons can be performed on these data types.
3. The list of data types (INTEGER, LONG, INT, SMALLINT) can store whole numbers. These data types are used for representing integer values. They differ in their range or the maximum value they can hold. "INTEGER" typically represents a 32-bit signed integer, "LONG" or "INT" represents a 64-bit signed integer, and "SMALLINT" represents a smaller range of signed integers. Arithmetic operations and numeric comparisons can be performed on these data types.
4. The list of data types (MONEY, CURRENCY) can store numeric values with decimal places specifically used for holding monetary values. These data types are designed to handle financial calculations and represent currency values. Arithmetic operations and numeric comparisons can be performed on these data types.
5. The list of data types (BINARY, LONGBINARY, GENERAL, IMAGE, OLEOBJECT) can store complete files, such as pictures or media. These data types are used for storing binary data or large objects. They can hold a large amount of data and are not of fixed size. These types are typically used for storing images, multimedia files, or any other binary data. Arithmetic operations and numeric comparisons are not applicable to these data types.
6. The list of data types (DATE, TIME, DATETIME) can store date and time values. These hybrid data types can represent specific dates, times, or both date and time combined. They can be used for performing arithmetic and numeric comparisons on dates and times, allowing calculations like adding or subtracting time intervals or comparing two dates. Although they are stored as text fields, they have built-in functionality for date and time operations.
7. CREATE TABLE Friend (

idno CHAR(3),

name CHAR(24),

address CHAR(24),

birthday DATE,

waistsize INTEGER,

giftvalue CURRENCY

);

1. CREATE TABLE Friend (

idno INT PRIMARY KEY,

name VARCHAR(50),

address VARCHAR(100),

age INT,

bankbalance DECIMAL(10, 2)

);

1. INSERT INTO Friend (name)

VALUES ('Road Runner');

1. Based on the previous question, the SQL command created a table named "Friend" with fields such as "idno," "name," "address," "birthday," "waistsize," and "giftvalue." However, since no specific values were provided for the fields other than the "name" field, the default value for each field would be NULL.

Therefore, for the record created by the previous question, the fields other than the "name" field would have NULL values.

11. INSERT INTO Friend (idno, name, address, birthday, waistsize, giftvalue)

VALUES ('123', 'Tasmanian Devil', 'Tasmania', '1957-07-07', 32, 29.99);

12.INSERT INTO Friend (idno, name, address, birthday, waistsize, giftvalue)

VALUES ('456', 'Felix the Cat', 'Hollywood', NULL, NULL, NULL);

1. INSERT INTO Friend (idno, name, address)

SELECT spno, name, addr

FROM Salesperson;

1. INSERT INTO Friend (idno, name, address)

SELECT custno, name, address

FROM Customer

WHERE state = 'WA';

1. INSERT INTO Friend (idno)

SELECT custno

FROM Customer

WHERE state = 'WA';

1. INSERT INTO Friend (idno, name)

SELECT spno, name

FROM Salesperson;

1. The issue with the given query is that it attempts to insert the values from the "spno," "name," and "commrate" columns of the "salesperson" table into the "Friend" table, specifically into the "idno," "name," and "giftvalue" columns, respectively.

The problem lies in the mapping of the "commrate" column to the "giftvalue" column. It doesn't make sense to insert the commission rate (commrate) into the gift value (giftvalue) column. These two columns likely represent different types of data and have different meanings. The commission rate is a percentage or a numeric value representing a percentage, while the gift value is expected to hold a monetary value.

To resolve this issue, either the "giftvalue" column in the "Friend" table should be changed to a column that makes sense for storing the commission rate, or a different appropriate column in the "Friend" table should be selected for mapping the commission rate.

1. The issue with the given query is that it attempts to insert all fields from the "Salesperson" table into the "Friend" table using the asterisk () wildcard. However, it's not possible to insert all fields if the "Friend" table doesn't have the same number of fields or if the respective fields don't match. The wildcard () can only be used when the number and types of columns in both tables match.
2. UPDATE Friend

SET giftvalue = 49.99

WHERE age >= 21;

1. UPDATE Friend

SET waistsize = 0;

1. ALTER TABLE Friend

ADD city CHAR(12);

1. ALTER TABLE Friend

ALTER COLUMN name CHAR(36);

1. ALTER TABLE Friend

DROP COLUMN city;

1. DELETE FROM Friend

WHERE giftvalue > 100;

1. DROP TABLE Friend;
2. The distinctive punctuation marks that should be used to enclose date values in queries depend on the database system being used. In some database systems, such as Microsoft SQL Server and Microsoft Access, the pound sign (#) is typically used to enclose date values. For example, #2023-05-23# represents May 23, 2023. However, different database systems may use different punctuation marks to enclose date values, so it's important to refer to the specific documentation of the database being used.
3. The different kinds of punctuation marks that can be used inside date values to separate month, day, and year depend on the database system as well. Commonly used separators include slashes (/), dashes (-), and periods (.), among others. For example, 2023/05/23, 2023-05-23, and 2023.05.23 all represent the date May 23, 2023. The specific punctuation mark used depends on the database system's date format and settings.
4. SELECT \*

FROM Carsale

WHERE salesdate > #01/01/2006#;

1. SELECT \*

FROM Carsale

WHERE salesdate > #01-01-06#;

1. SELECT FORMAT(salesdate, 'MM/dd/yyyy') AS formatted\_salesdate

FROM Carsale;

1. SELECT FORMAT(salesdate, 'dddd, mmmm dd, yyyy') AS formatted\_salesdate

FROM Carsale;

1. SELECT FORMAT(salesdate, 'dd MMM yy') AS formatted\_salesdate

FROM Carsale;

1. SELECT FORMAT(salesdate, 'yyyy, ww, d') AS formatted\_salesdate

FROM Carsale;