CS669

SiCheng Yi

LAB2

**Section One – Relating Data**

1. Create the Pizza and Toppings tables, including all of their columns, datatypes, and constraints, including the foreign key constraint.

CREATE TABLE Pizza(

pizza\_id DECIMAL(12) NOT NULL PRIMARY KEY,

name VARCHAR(32) NOT NULL,

date\_available DATE NOT NULL,

price DECIMAL(4,2) NOT NULL

);

CREATE TABLE Topping(

topping\_id DECIMAL(12) NOT NULL PRIMARY KEY,

topping\_name VARCHAR(64) NOT NULL,

pizza\_id DECIMAL(12) NULL

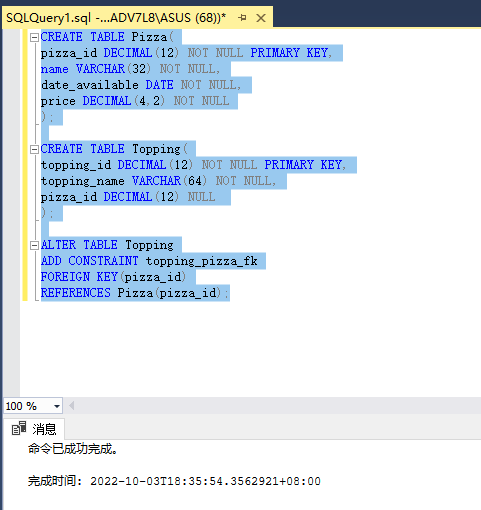
);

ALTER TABLE Topping

ADD CONSTRAINT topping\_pizza\_fk

FOREIGN KEY(pizza\_id)

REFERENCES Pizza(pizza\_id);



2. Insert at least four rows into the Pizza table. One of the pizzas should be named “Plain” because it will not have any toppings. Other than this, you select the ids, names, dates, and prices of your choosing (maybe you have some favorite pizzas?).

Insert toppings of your choosing into the topping table. One of the toppings should not be associated with any pizza, that is, the topping should be an “add-on” which is not included in any pizza’s standard toppings. Ensure that the “Plain” pizza has no toppings, and that all other pizzas have at least two toppings. Select all rows in both tables to view what you inserted.

INSERT INTO Pizza(pizza\_id, name, date\_available, price)

VALUES(1, 'Plain', CAST('3/10/2022' AS date), 8.00);

INSERT INTO Pizza(pizza\_id, name, date\_available, price)

VALUES(2, 'Hawaiian', CAST('4/10/2022' AS date), 10.12);

INSERT INTO Pizza(pizza\_id, name, date\_available, price)

VALUES(3, 'Margherita', CAST('5/10/2022' AS date), 12.25);

INSERT INTO Pizza(pizza\_id, name, date\_available, price)

VALUES(4, 'Supreme', CAST('6/10/2022' AS date), 18.75);

INSERT INTO Topping(topping\_id, topping\_name, pizza\_id)

VALUES(101, 'pineapple', 2);

INSERT INTO Topping(topping\_id, topping\_name, pizza\_id)

VALUES(102, 'Bacon', 2);

INSERT INTO Topping(topping\_id, topping\_name, pizza\_id)

VALUES(103, 'pork', NULL);

INSERT INTO Topping(topping\_id, topping\_name, pizza\_id)

VALUES(104, 'tomato', 3);

INSERT INTO Topping(topping\_id, topping\_name, pizza\_id)

VALUES(105, 'cheese', 3);

INSERT INTO Topping(topping\_id, topping\_name, pizza\_id)

VALUES(106, 'sausage', 4);

INSERT INTO Topping(topping\_id, topping\_name, pizza\_id)

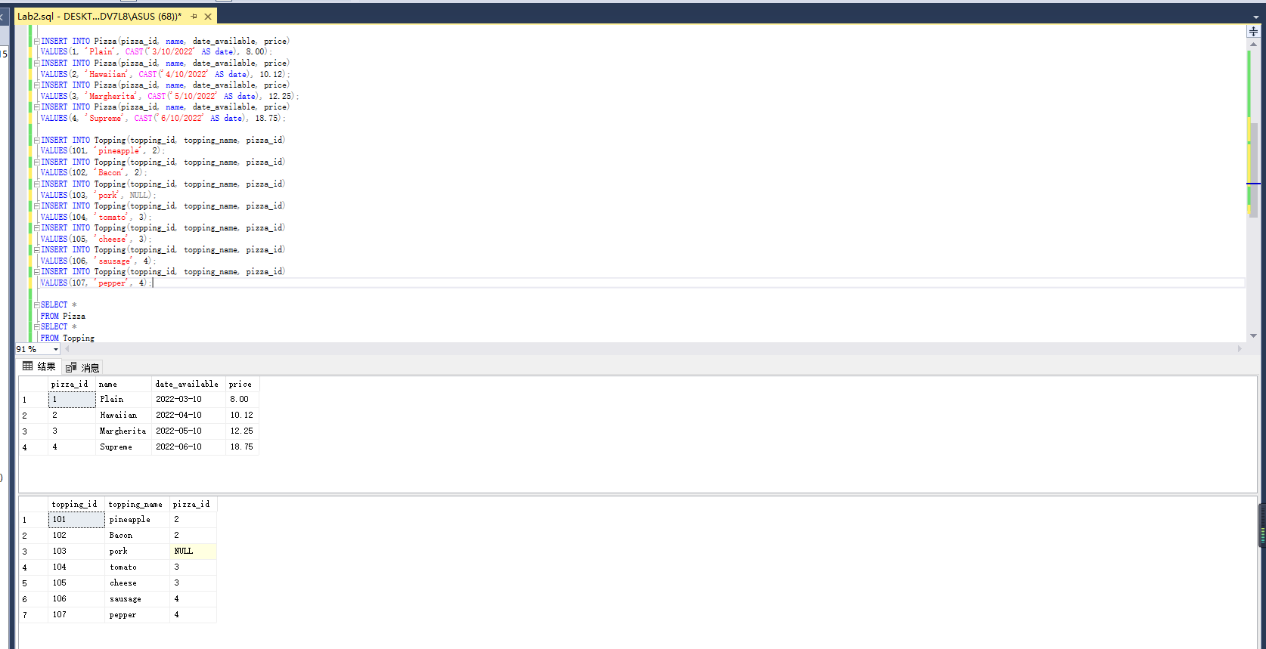
VALUES(107, 'pepper', 4);

SELECT \*

FROM Pizza

SELECT \*

FROM Topping



3. As an exercise, attempt to insert a topping that references a pizza that doesn’t exist. Summarize:

a. why the insertion failed, and

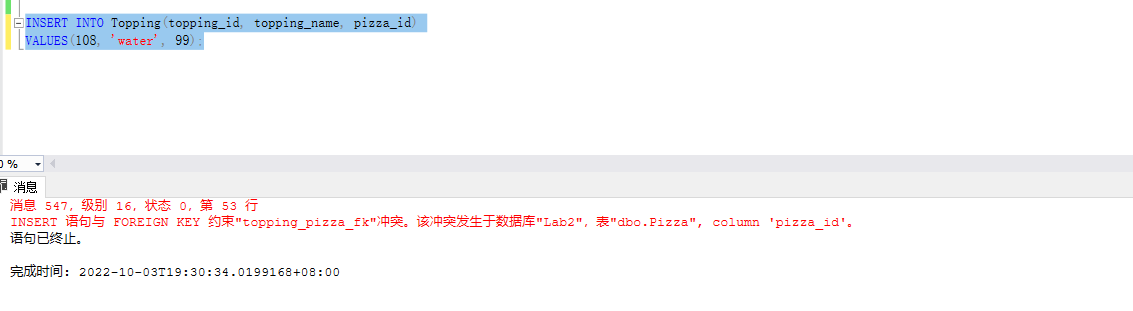
b. how you would interpret the error message from your RDBMS so that you know that the error indicates the Pizza reference is invalid.

INSERT INTO Topping(topping\_id, topping\_name, pizza\_id)

VALUES(108, 'water', 99);

a:According to the error message, we can easily judge that the reason for the failure is because of a FOREIGN KEY conflict, because we inserted a pizza\_id (FOREIGN KEY) that does not exist.

b:The error message tells us that the reason for the failure is because of the FOREIGN KEY conflict, and also tells us that the conflict occurred in the database "Lab2", the table "dbo.Pizza", the column 'pizza\_id', so we need to confirm whether the data in the changed location is correct, We will find that the pizza\_id is invalid.



4. Summarize:

a. what a join is and how joins help answer questions using related data, and

b. the similarities and differences between an inner join, a left join, a right join, and a full outer join.

a.A join is both a fundamental concept detailing how data from related tables can be retrieved, as well as a critical SQL operation. It then uses related data to present client a more complete picture of the information that were separated in different tables.

Join cartesian product is created, then the matching rows are selected from that product. These rows are then combined to answer the Retrieving Related Data question.

b. The similarities is that they are all creating the cartesian product and all four keep the rows that match the join condition.

The differences is that:

Inner join keeps only eligible rows and deletes all other rows.

Left join keeps the rows that match the join condition as well a distinct list of rows from the leftmost table that did not match the join condition.

Right join keeps the rows that match the join condition as well a distinct list of rows from the rightmost table that did not match the join condition.

Outer join keeps the rows that match the join condition are included, as well as a distinct list of rows from both the left and right tables that did not match the join condition.

5. With a single SQL query, fulfill the following request:

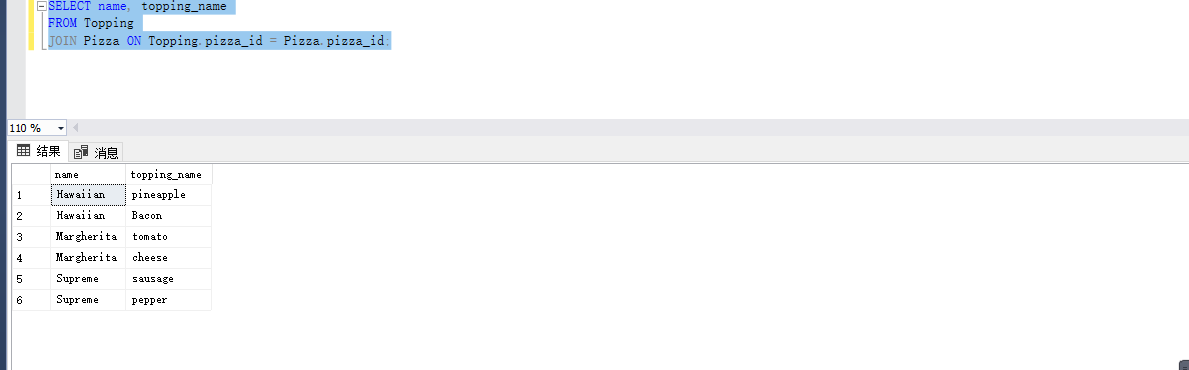
List the names of the pizzas that have toppings, and the names of all of the toppings that go with each pizza.

From a technical SQL perspective, explain why some rows in the Pizza table and some rows in the Toppings table were not listed.

SELECT name, topping\_name

FROM Topping

JOIN Pizza ON Topping.pizza\_id = Pizza.pizza\_id;



Here I use inner join, so SQL will create the cartesian product and keep the rows that match the join condition, then delete all other rows. So it does not include pizza without topping and topping without corresponding pizza.

6. Fulfill the following request:

List the names and availability date of all pizzas whether or not they have toppings. For the pizzas that have toppings, list the names of the toppings that go with each of those pizzas. Order the list by the availability date, oldest to newest.

There are two kinds of joins that can be used to satisfy this request. Write two queries using each type of join to satisfy this request.

SELECT name, date\_available, topping\_name

From Topping

RIGHT JOIN Pizza ON Topping.pizza\_id = Pizza.pizza\_id

ORDER BY date\_available;

SELECT name, date\_available, topping\_name

From Topping

FULL JOIN Pizza ON Topping.pizza\_id = Pizza.pizza\_id

ORDER BY date\_available;

图形用户界面, 应用程序

描述已自动生成

7. Fulfill the following request:

List the names of all toppings whether or not they go with a pizza, and the names of the pizzas the toppings go with. Order the list by topping name in reverse alphabetical order.

Just as with step #6, there are two kinds of joins that can be used to satisfy this request. Write two queries using each type of join to satisfy this request.

SELECT topping\_name, name

From Topping

LEFT JOIN Pizza ON Topping.pizza\_id = Pizza.pizza\_id

ORDER BY topping\_name DESC;

SELECT topping\_name, name

From Pizza

RIGHT JOIN Topping ON Pizza.pizza\_id = Topping.pizza\_id

ORDER BY topping\_name DESC;

图形用户界面, 应用程序

描述已自动生成

8. Fulfill the following request with a single SQL query:

List the names of all pizzas and all toppings, as well as which pizzas go with which toppings. Order the list alphabetically by pizza name then by topping name.

SELECT name, topping\_name

From Topping

FULL JOIN Pizza ON Topping.pizza\_id = Pizza.pizza\_id

ORDER BY name, topping\_name;

图形用户界面, 应用程序, Word

描述已自动生成

**Section Two – Expressing Data**

9. Summarize in your own words why it’s not ideal to manually lookup or eyeball answers to questions about data in a database.

The execution speed of the system is much faster than that of humans. When the data becomes large or needs to be executed for a long time, humans need to spend a lot of energy and time to complete the goal, and the accuracy of execution is far less than that of AI.

10. Summarize in your own words four significant components that determine how a value is displayed in your SQL client.

They are Raw Values，Manipulations on the Value， Formatting Constructs and SQL Client Choices.

Raw Values: Initial settings for the size and specification of the data.

Manipulations on the Value: The size and specification set after processing the data.

Formatting Constructs: Format data uniformly into a set structure.

SQL Client Choices: The choice of database client affects the structure and rules of the data.

11. Summarize in your own words:

a. what expressions are.

b. what operator precedence is and the role it plays in expressions.

c. what datatype precedence is and the role it plays in expressions.

a: Operations that affect the way the data is displayed in the SQL query.

b: The order of precedence of each operator is called operator precedence, which determines the order of operations in an expression.

c: The precedence order of data types is called datatype precedence, and it determines the data type that the expression will produce.

12. Fulfill the following request with a single query:

Management of the pizza shop wants to review its pizza pricing. List the names and prices of all pizzas, making sure to format the price monetarily in U.S. dollars (for example, “$11.99”).

SELECT name, format(price, '$.00')

FROM Pizza

图形用户界面, 应用程序, Word

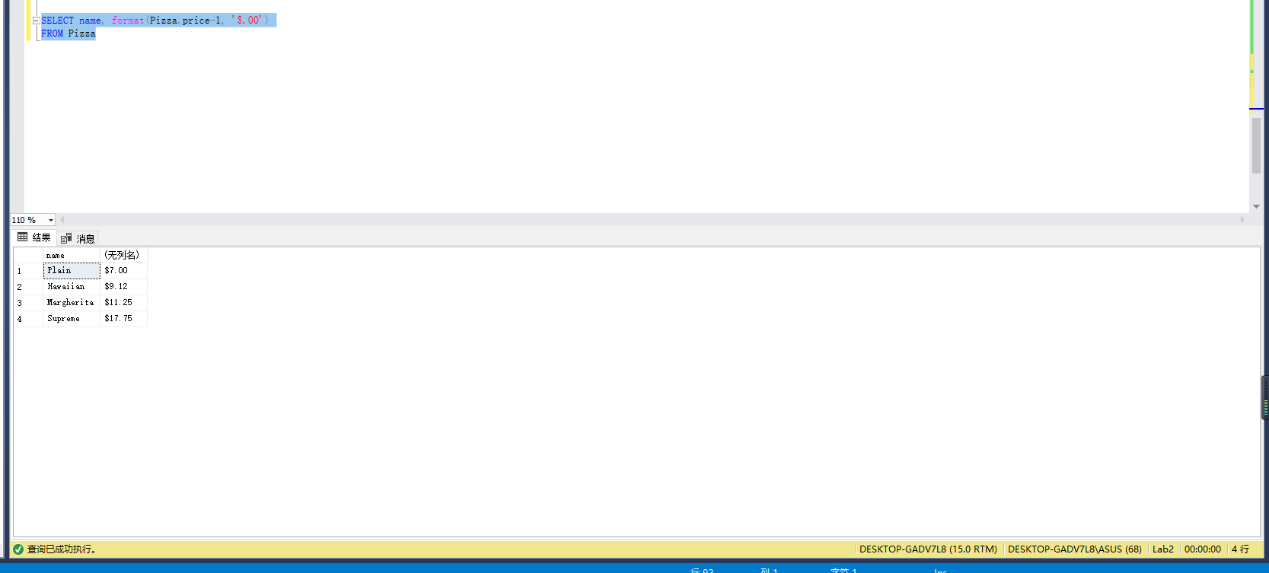
描述已自动生成

13. Fulfill the following request with a single query:

The pizza shop is running a special where every pizza is discounted by a dollar. List the names and discounted prices of all pizzas, making sure to format the price monetarily in U.S. dollars.

SELECT name, format(Pizza.price-1, '$.00')

FROM Pizza



14. Fulfill the following request with a single query:

The pizza shop wants to mail out mailers that promotes all of the toppings it offers, tied into the pizzas it sells. The shop wants each line in the mailer formatted like “ToppingName (PizzaName: Price)”, and wants the lines ordered alphabetically by topping name. For example, if a “Meat Lover’s” pizza costs $10.00 and has two toppings – Sausage and Pepperoni – the results would have two lines for this pizza:

Pepperoni (Meat Lover’s: $10.00)

Sausage (Meat Lover’s: $10.00)

SELECT topping.topping\_name + '(' + Pizza.name + ':' + format(Pizza.price, '$.00') + ')'

FROM Topping

JOIN Pizza ON Topping.pizza\_id = Pizza.pizza\_id

ORDER BY topping\_name;

