# Creating Horizontal Partitions in SQL - Finalizing Data Insertion

## Question:

In this exercise, you need to create a table `film\_partitioned`, partitioned on the field `release\_year` using \*\*list partitioning\*\*. Then, create three partitions for the years \*\*2017, 2018, and 2019\*\* with table names `film\_2017`, `film\_2018`, and `film\_2019`, respectively. Finally, insert the data into `film\_partitioned` and query the partitioned table.

## Full Answer (SQL Code):

-- Create a new table called film\_partitioned  
CREATE TABLE film\_partitioned (  
 film\_id INT,  
 title TEXT NOT NULL,  
 release\_year TEXT  
)  
PARTITION BY LIST (release\_year);  
  
-- Create the partitions for 2019, 2018, and 2017  
CREATE TABLE film\_2019  
 PARTITION OF film\_partitioned FOR VALUES IN ('2019');  
  
CREATE TABLE film\_2018  
 PARTITION OF film\_partitioned FOR VALUES IN ('2018');  
  
CREATE TABLE film\_2017  
 PARTITION OF film\_partitioned FOR VALUES IN ('2017');  
  
-- Insert the data into film\_partitioned  
INSERT INTO film\_partitioned (film\_id, title, release\_year)  
SELECT film\_id, title, release\_year FROM film;  
  
-- View film\_partitioned  
SELECT \* FROM film\_partitioned;

## Explanation of the Answer:

The `CREATE TABLE` statement first defines `film\_partitioned` as a partitioned table using `LIST` partitioning on `release\_year`. Each partition (`film\_2019`, `film\_2018`, `film\_2017`) is created separately to store data for specific years. The `INSERT INTO ... SELECT` command transfers data from the `film` table to the partitioned table. Finally, the `SELECT \* FROM film\_partitioned;` statement retrieves all records, allowing verification of successful partitioning.