HONG KONG INSTITUTE OF VOCATIONAL EDUCATION

## Laboratory 3: Database User Defined Function, Event, View, Trigger

**Module Intended Learning Outcome (#3):**

On completion of the module, students are expected to be able to:

* Develop database queries to perform typical data definition and data manipulation operations in relational database systems.

## Lesson Intended Learning Outcome:

On completion of this lab, students are expected to be able to:

* use Navicat to create and utilize User Defined Functions (UDFs), Events, Views, and Triggers in MySQL

**Theme Park database**

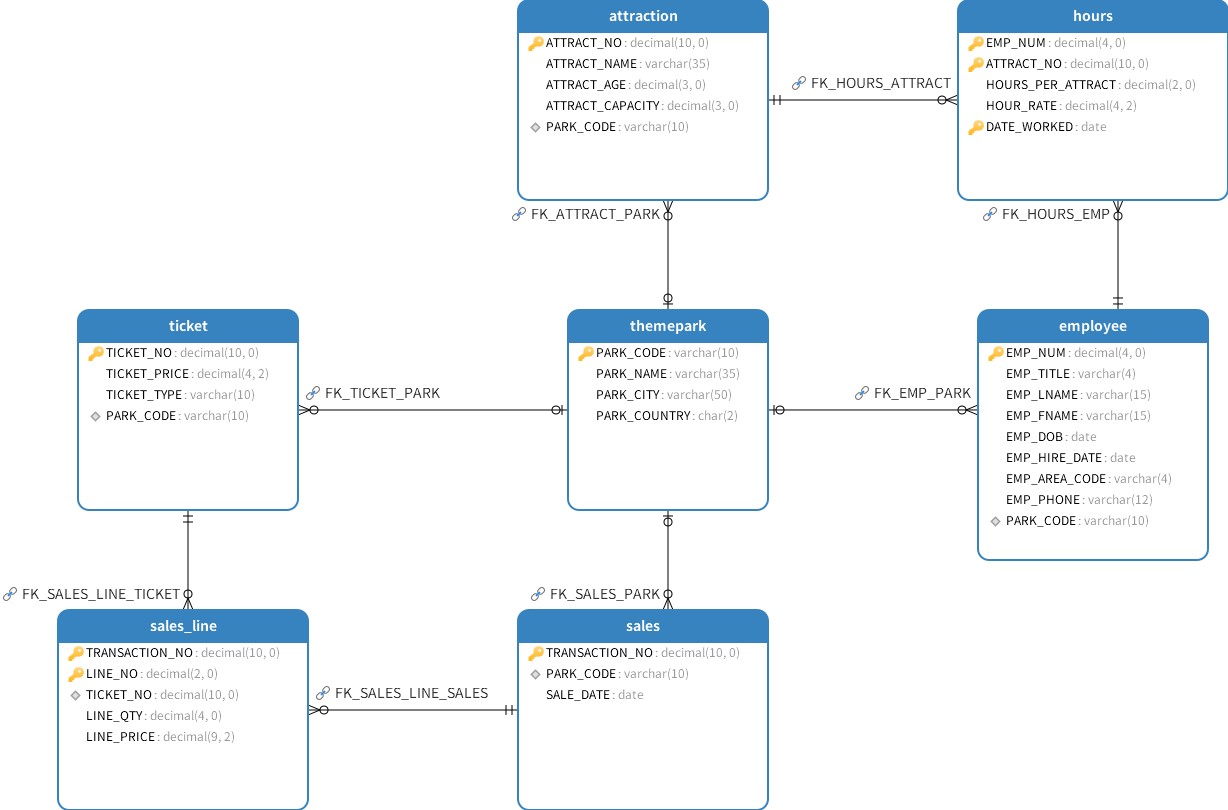
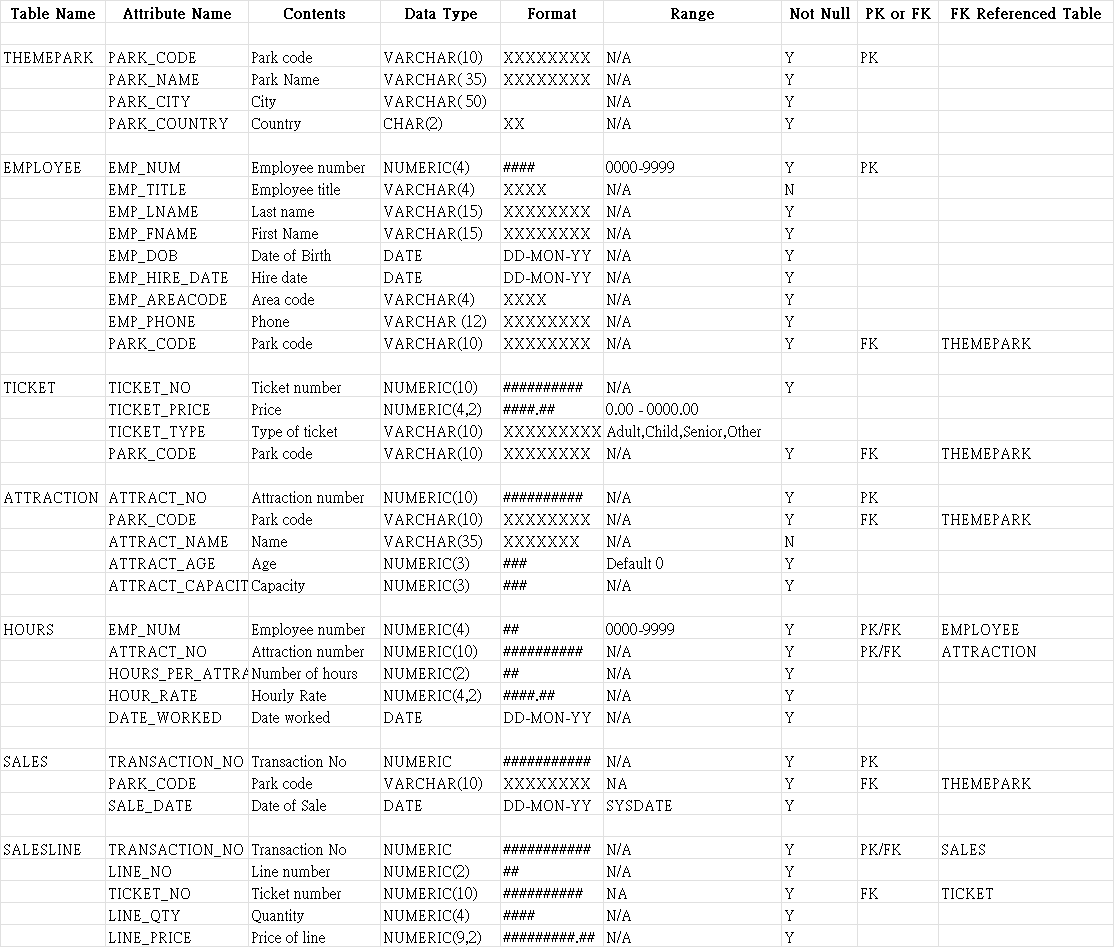
****

Figure 1 Entity Relationship Diagram



Data Dictionary of Theme Park database

**Basic Functions:**

|  |
| --- |
| **CURDATE()**: Returns the current date.  **YEAR(date)**: Extracts the year part from a date.  **DATE\_SUB(date, INTERVAL value unit)**: Subtracts a specified time period (e.g., days, months, years) from a date.  **SIGNAL SQLSTATE '45000'**: Generates a custom error message.  **SET MESSAGE\_TEXT = 'message'**: Sets the custom error message text.  **TIMESTAMPDIFF(unit,datetime1,datetime2)**: calculates the difference between two date or datetime values, returning the result in the specified unit of time. For example: TIMESTAMPDIFF( MONTH, ‘2009-05-18’,’2009-07-29’) |

# TASK:

**User Defined Function**

**Concept: A User Defined Function (UDF) in SQL is a reusable code block that performs a specific task, such as a calculation, and returns a value. It can take input parameters and perform operations using those inputs.**

**Task 1.1:** Create a function calculate\_age that takes an employee's date of birth (DOB) as an input and returns the current age of the employee.

* **Instructions:** Use the TIMESTAMPDIFF function to compute the difference in years between the DOB and the current date.
* **Example:** calculate\_age('1972-06-15') should return 52 (as of 2024).

|  |
| --- |
| DELIMITER //  CREATE FUNCTION calculate\_age(dob DATE) RETURNS INT  BEGIN  DECLARE age INT;  SET age = TIMESTAMPDIFF(YEAR, dob, CURDATE());  RETURN age;  END //  DELIMITER ;  SELECT EMP\_FNAME, EMP\_LNAME, calculate\_age(EMP\_DOB) AS Age  FROM EMPLOYEE; |

**Scheduled Event**

**Concept: An Event in SQL is a scheduled task that runs automatically at a specific time or interval. Events are useful for performing regular database maintenance or automation tasks.**

**Task 2.1:** Create an event increase\_ticket\_prices to increase all ticket prices by 5% every year on January 1st.

* **Instructions:** Use the ON SCHEDULE EVERY 1 YEAR clause to set the event to run annually. Update the TICKET table to increase prices.
* **Example:** If a ticket price is 20.00, it should become 21.00 after the event is executed.

|  |
| --- |
| CREATE EVENT increase\_ticket\_prices  ON SCHEDULE EVERY 1 YEAR STARTS '2024-01-01 00:00:00'  DO  UPDATE TICKET SET TICKET\_PRICE = TICKET\_PRICE \* 1.05; |

**View**

**Concept: A View in SQL is a virtual table that contains the result of a query. Views can simplify complex queries and provide a security layer by restricting direct access to table data.**

**Task 3.1:** Create a view **Employee\_Details** that shows employee number, full name, combined telephone number (area code and phone number). and park name for each employee.

The view should include the following columns:

* EMP\_NUM: Employee number
* Full\_Name: Combines the first and last name of the employee.
* ContactNumber: This should be a concatenation of the EMP\_AREA\_CODE and EMP\_PHONE columns, formatted as area code-phone number.

For example, if an employee's EMP\_AREA\_CODE is 0181 and their EMP\_PHONE is 324-9134, the ContactNumber should be displayed as 0181-324-9134.

|  |
| --- |
| CREATE VIEW Employee\_Details AS  SELECT EMP\_NUM,  CONCAT(EMP\_FNAME, ' ', EMP\_LNAME) AS Full\_Name,  CONCAT(EMP\_AREA\_CODE, '-', EMP\_PHONE)  PARK\_NAME  FROM EMPLOYEE  JOIN THEMEPARK ON EMPLOYEE.PARK\_CODE = THEMEPARK.PARK\_CODE; |

**Triggers**

**Concept: A Trigger in SQL is a piece of code that automatically executes in response to certain events on a table, such as INSERT, UPDATE, or DELETE. Triggers can enforce business rules and maintain data integrity.**

**Task 4.1:** Create a trigger `ticket\_price\_update` that logs changes to ticket prices into a Ticket\_Log table before any update to the TICKET table.

**Instructions:**

1. Create the `Ticket\_Log` Table:

|  |
| --- |
| CREATE TABLE Ticket\_Log (  LOG\_ID INT AUTO\_INCREMENT PRIMARY KEY,  TICKET\_NO NUMERIC(10),  OLD\_PRICE NUMERIC(4,2),  NEW\_PRICE NUMERIC(4,2),  CHANGE\_DATE TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  ); |

1. Create the Trigger and the definition as shown below:

|  |
| --- |
| INSERT INTO Ticket\_Log (TICKET\_NO, OLD\_PRICE, NEW\_PRICE)  VALUES (OLD.TICKET\_NO, OLD.TICKET\_PRICE, NEW.TICKET\_PRICE); |

1. Test the trigger by updating some ticket prices and then querying the Ticket\_Log table to verify the changes were logged.

|  |
| --- |
| UPDATE TICKET SET TICKET\_PRICE = 29.99 WHERE TICKET\_NO = 89720;  SELECT \* FROM Ticket\_Log; |

**Example:** If ticket 89720 has its price changed from 24.99 to 29.99, this change should be recorded in the Ticket\_Log.

**Task 4.2:** Create a trigger BeforeInsertEmployee that checks if the employee is at least 18 years old before inserting their record.

**Instructions:**

1. Create the Trigger and the definition as shown below:

|  |
| --- |
| IF YEAR(CURDATE()) - YEAR(NEW.EMP\_DOB) < 18 THEN  SIGNAL SQLSTATE '45000'  SET MESSAGE\_TEXT = 'Employee must be at least 18 years old';  END IF; |

1. Test this trigger by trying to insert an employee under 18 years old.

**Example:**

If a new employee is 16 years old, the insertion should fail with an error message.

**Task 4.3:** Create a trigger BeforeDeleteSales that prevents deleting sales records if the sales date is less than a month old.

**Instructions:**

1. Create the Trigger and the definition as shown below:

|  |
| --- |
| IF OLD.SALE\_DATE > DATE\_SUB(CURDATE(), INTERVAL 1 MONTH) THEN  SIGNAL SQLSTATE '45000'  SET MESSAGE\_TEXT = 'Cannot delete sales records less than a month old';  END IF; |

1. Create a new sale record into SALES Table.
2. Attempt to delete a recent sale and observe the error.

**Example:**

If the sales date is less than a month old, the deletion should fail with an error message.

END