MySQL handler example in stored procedures

First, create a new table named **SupplierProducts** for the demonstration:

```
CREATE TABLE SupplierProducts
(
   supplierId INT,
   productId INT,
   PRIMARY KEY (supplierId , productId)
);
```

The table SupplierProducts stores the relationships between the table suppliers and products. Each supplier may provide many products and each product can be provided by many suppliers. For the sake of simplicity, we don’t create Products and Suppliers tables, as well as the foreign keys in the SupplierProducts table.

Second, create a stored procedure that inserts product id and supplier id into the SupplierProducts table:

```
CREATE PROCEDURE InsertSupplierProduct(IN inSupplierId INT, IN inProductId
INT)
BEGIN
   -- exit if the duplicate key occurs
   DECLARE EXIT HANDLER FOR 1062
   BEGIN
           SELECT ‘Duplicate value’ AS message;
   END;

   -- insert a new row into the SupplierProducts
   INSERT INTO SupplierProducts(supplierId,productId)
   VALUES(inSupplierId,inProductId);

   -- return the products supplied by the supplier id
   SELECT COUNT(*) FROM SupplierProducts WHERE supplierId = inSupplierId;
   END$$

DELIMITER ;
```

How it works.

The following exit handler terminates the stored procedure whenever a duplicate key occurs (with code 1062). In addition, it returns an error message.

```
DECLARE EXIT HANDLER FOR 1062BEGIN
   SELECT ‘Duplicate value’ AS message;END;
```

This statement inserts a row into the SupplierProducts table. If a duplicate key occurs, the code in the handler section will execute.

```
INSERT INTO SupplierProducts(supplierId,productId)
VALUES(supplierId,productId);
```

Third, call the InsertSupplierProduct() to insert some rows into the SupplierProducts table:

```
CALL InsertSupplierProduct(1,1);
CALL InsertSupplierProduct(1,2);
CALL InsertSupplierProduct(1,3);
```

Fourth, attempt to insert a row whose values already exist in the SupplierProducts table:

```
CALL InsertSupplierProduct(1,3);
```

Here is the error message:

```
+------------------------------+
| message                      |
+------------------------------+
| Duplicate key (1,3) occurred |
+------------------------------+1 row in set (0.01 sec)Code language: JavaScript
(javascript)
```

Because the handler is an EXIT handler, the last statement does not execute:

```
SELECT COUNT(*) FROM SupplierProductsWHERE supplierId = inSupplierId;
```

If you change the EXIT in the handler declaration to CONTINUE , you will also get the number of products provided by the supplier:

```
DROP PROCEDURE IF EXISTS InsertSupplierProduct;

DELIMITER $$
CREATE PROCEDURE InsertSupplierProduct(IN inSupplierId INT,
   IN inProductId INT)
   BEGIN

   DECLARE CONTINUE HANDLER FOR 1062
   BEGIN
           SELECT 'Duplicate value' AS message;
   END;

   -- insert a new row into the SupplierProducts
   INSERT INTO SupplierProducts(supplierId,productId)
   VALUES(inSupplierId,inProductId);

   -- return the products supplied by the supplier id
   SELECT COUNT(*) FROM SupplierProducts
   WHERE supplierId = inSupplierId;
```

```
    END$$

DELIMITER ;
```

Finally, call the stored procedure again to see the effect of the CONTINUE handler:

```
CALL InsertSupplierProduct(1,3);
```

Here is the output:

```
+----------+
| COUNT(*) |
+----------+
|        3 |
+----------+
1 row in set (0.01 sec)
Query OK, 0 rows affected (0.02 sec)
```

List of Exceptions

Exception Number	**Exception Name**	**Detail**
1146	Table does't exist	Table is not present in MySQL
1040	Too Many Connections	MySQL reaches maximum number of client connections.
1045	Access denied	Access denied to perform DML operation due to user doesn't exist or password is incorrect.
1064	Syntax error	Mistyped commands
1114	Table is full	Unable to insert in table due to disk is full.
2006	Connection closed	MySQL server connection timed out.