Project API

This documentation describes stored procedures that are part of the *application programming interface (API)* provided by the project for database applications to call. We divide all the stored procedures into two parts: merchant oriented and customer oriented.

Merchant Oriented

1. **createProduct**(*certainName varchar(100), certainDescription varchar(2000), certainSKU varchar(12)*)

   Create a new product with given name, description and SKU.

   Throw an error message if encounters duplicate primary key, invalid SKU or empty name.

   Throw a SQLException if any other error occurs.

2. **listProduct**(*cartainSKU varchar(12)*)

   List a product on the website to make it available for customers, and add current price information into the PriceHistory table.

   Throw an error message if the product has already been listed, the product cannot be found or the price has not been set.

3. **unlistProduct**(*cartainSKU varchar(12)*)

   Unlist a product from the website.

   Throw an error message if the product has already been unlisted or the product cannot be found.

4. **readProductInventory**(*isListed boolean*)

   Display all the listed/unlisted product information in table Product and InventoryRecord.

5. **readAllProductInventory**()

   Display all the product information in table Product and InventoryRecord.

6. **readInventorySpecifyingUnit**(*beginUnit int, endUnit int*)

   Display all the product information in table Product and InventoryRecord for a specific units range.

   Throw an error message if beginUnit is larger than endUnit.

   Throw a "Not Found" message if the query output is null.

7. **readSpecificProuductInventory**(*certainSKU varchar(12)*)

   Display inventory record for a specific product.

Throw a "Not Found" message if the query output is null.

8. **changeInventoryUnits**(*certainSKU varchar(12), inputUnits int*)

   Change inventory units of a product.

   Throw an error message if the inventory units after change is less than 0 or the product cannot been found.

9. **changePreOrderUnits**(*certainSKU varchar(12), inputPreOrderUnit int*)

   Change pre-order units of a product.

   Throw an error message if the pre-order units after change is less than 0 or the product cannot be found.

10. **changePrice**(*certainSKU varchar(12), certainPrice double(40, 2)*)

    Change price of a product and add the new sale price into PriceHistory table.

    Throw an error message if the price after change is less than 0 or the product cannot be found.

11. **changeDiscount**(*certainSKU varchar(12), certainDiscount double(40, 2)*)

    Change discount of a product and add the new sale price into PriceHistory table.

    Throw an error message if the discount after change is not between 0 and 1 or the product cannot be found.

12. **calculateTotalRevenue**(*startDate datetime, endDate datetime*)

    Calculate total revenue for a given time range.

    Throw an error message if startDate is later than endDate.

Customer Oriented

1. **createCustomer**(*inputCustomerID int, inputName varchar(20), inputAddress varchar(20), inputCity varchar(20), inputState varchar(20), inputCountry varchar(20), inputPostalCode varchar(10)*)

   Create a new customer with given customerID, name, address, city, state, country and postal code.

   Throw an error message if encounters duplicate primary key or invalid input country or state.

   Throw a SQLException if any other error occurs.

2. **readListedProducts**()

   Display all the listed product names, descriptions, prices, discounts and sale prices to customers.

   Throw a "Not Found" message if the query output is null.

3. **searchProductName**(*search varchar(100)*)

   Search listed products that contains the input string in their names, and display relevant product information.

Throw a "Not Found" message if the query output is null.

4. **readPriceHistory**(*certainSKU varchar(12)*)

   Display all the price history for a given product.

   Throw a "Not Found" message if the query output is null.

5. **readPriceHistorySpecifyingTime**(*certainSKU varchar(12), startTime datetime, endTime datetime*)

   Display all the price history for a given product and specified time range.

   Throw an error message if startDate is later than endDate.

   Throw a "Not Found" message if the query output is null.

6. **createOrder**(*certainCustomerID int, certainOrderID varchar(40), certainDate datetime*)

   Create a new order with given customer ID, order ID and order date.

   Throw an error message if encounters duplicate primary key or a foreign key constraint fails.

   Throw a SQLException if any other error occurs.

7. **updateShipmentDate**(*certainOrderID varchar(40), certainShipmentDate*)

   Update shipment date for an order.

   Throw an error message if the input shipment date is earlier than order date or the order ID cannot be found.

8. **createOrderRecord**(*certainSKU varchar(12), certainOrderID varchar(40), certainUnits int*)

   Create an order record for a certain product in an order. When the order record is created, the price is calculated according to the price and discount in the InventoryRecord table and inventory is automatically reduced. Since we have Units and preOrderUnits, Units is deducted first. When Units becomes 0, preOrderUnits is used and PreOrderUnitsSold is recorded for future reference.

   Throw an error message and rollback if certainUnits is negative, certainUnits exceeds inventory or any other error happens.

9. **calculateOrderPrice**(*certainOrderID varchar(40)*)

   Calculate total price for a given order ID

   Throw an error message if the order ID cannot be found.

10. **finalizeOrderPrice**(*certainOrderID varchar(40)*)

    Finalize total price for a given order ID after considering promotion amount and tax.

    Throw an error message if the order ID cannot be found.

11. **createOneOrderWithManyItems**(*certainCustomerID int, certainOrderID varchar(40), certainOrderDate datetime,sku1 varchar(12), units1 int,sku2 varchar(12), units2 int,sku3 varchar(12), units3 int*)

    An aggregate simulating procedure for placing an order. Assuming only three products can be purchased in one order.

12. **cancelOrder**(*certainOrderID varchar(40)*)

    Cancel an order.

    Throw an error message if the order ID cannot be found.

13. **readCustomerOrders**(*certainCustomerID int*)

    Display all the orders for a given customer ID.

    Throw an error message if the customer ID cannot be found.

14. **readCustomerOrdersSpecifyingTime**(*certainCustomerID int, startDate datetime, endDate datetime*)

    Display all the orders for a given customer ID and time range.

    Throw an error message if startDate is later than endDate or the customer ID cannot be found.

15. **readOrderDetail**(*certainOrderID varchar(40)*)

    Display all the product names, prices and units bought for a given order ID.

    Throw an error message if the order ID cannot be found.

Triggers

1. **applyPromotion**

   Promotion trigger: get $50 off when spending $300+

   If the total price of an order is no less than $300, add the order ID and promotion amount into the Promotion table.

2. **unlistProductIfNotAvailable**

   Unlist a product if Units and PreOrderUnits in the InventoryRecord table both become zero.