**Name: Jianwei Su**

**Date:01/07/2023**

# **Exercises**

## **Level 7: Case Study**

### **7.1: ADVANCED QUERIES**

1. Display total market value traded for fund Whale Rock on each day in Jan 2018.

**CREATE TEMPORARY TABLE TEMP(**

**SELECT TDH.COB\_DATE, TDH.CUSIP,TDH.FUND\_ID,FI.FUND\_NAME,TDH.QUANTITY**

**FROM trade\_data\_hist TDH**

**INNER JOIN fund\_info FI**

**ON TDH.FUND\_ID=FI.FUND\_ID AND TDH.COB\_DATE BETWEEN FI.START\_DATE AND FI.END\_DATE**

**WHERE FI.FUND\_NAME='Whale Rock' AND TDH.COB\_DATE<=20180131**

**);**

**SELECT TEMP.COB\_DATE,TEMP.FUND\_ID, TEMP.FUND\_NAME,**

**SUM(TEMP.QUANTITY\*SPH.CLOSE\_PRICE\_USD) Total\_Market\_Value\_Each\_day\_Jan\_2018**

**FROM TEMP**

**INNER JOIN security\_price\_hist SPH**

**ON TEMP.COB\_DATE=SPH.COB\_DATE AND TEMP.CUSIP=SPH.CUSIP**

**GROUP BY TEMP.COB\_DATE**

**ORDER BY TEMP.COB\_DATE ASC;**

**Table

Description automatically generated**

1. Which ticker contained the highest avg daily market value for a single fund over the history of our dataset, and which ticker & fund was it?

**CREATE TEMPORARY TABLE TEMP1(**

**SELECT TDH.COB\_DATE, TDH.CUSIP,TDH.FUND\_ID,FI.FUND\_NAME,TDH.QUANTITY**

**FROM trade\_data\_hist TDH**

**INNER JOIN fund\_info FI**

**ON TDH.FUND\_ID=FI.FUND\_ID AND TDH.COB\_DATE BETWEEN FI.START\_DATE AND FI.END\_DATE**

**);**

**CREATE TEMPORARY TABLE TEMP2(**

**SELECT TEMP1.COB\_DATE, TEMP1.CUSIP, TEMP1.FUND\_ID, TEMP1.FUND\_NAME, TEMP1.QUANTITY, SPH.CLOSE\_PRICE\_USD**

**FROM TEMP1**

**INNER JOIN security\_price\_hist SPH**

**ON TEMP1.COB\_DATE=SPH.COB\_DATE AND TEMP1.CUSIP=SPH.CUSIP**

**ORDER BY TEMP1.FUND\_ID,TEMP1.COB\_DATE);**

**SELECT TEMP2.FUND\_ID, TEMP2.FUND\_NAME, SI.TICKER,AVG(TEMP2.QUANTITY\*TEMP2.CLOSE\_PRICE\_USD) AVG\_Daily\_Market\_Value**

**FROM TEMP2**

**INNER JOIN security\_info SI**

**ON TEMP2.CUSIP=SI.CUSIP AND TEMP2.COB\_DATE BETWEEN SI.START\_DATE AND SI.END\_DATE**

**GROUP BY SI.TICKER, TEMP2.FUND\_NAME**

**ORDER BY AVG\_Daily\_Market\_Value DESC;**

**Graphical user interface

Description automatically generated with medium confidence**

**Based on the result above, Ticker ADP and Fund Tree Free contained the highest avg daily market value for a single fund over the history of our dataset.**

1. Which region has the largest combined VaR for the month of March 2018?

**CREATE TEMPORARY TABLE TEMP1(**

**SELECT TDH.COB\_DATE,TDH.CUSIP,RDH.VAR**

**FROM trade\_data\_hist TDH**

**INNER JOIN risk\_data\_hist RDH**

**ON TDH.CUSIP=RDH.CUSIP AND TDH.POSITION\_ID=RDH.POSITION\_ID**

**WHERE TDH.CUSIP<>'' AND TDH.COB\_DATE BETWEEN 20180301 AND 20180331**

**);**

**CREATE TEMPORARY TABLE TEMP2(**

**SELECT TEMP1.COB\_DATE, TEMP1.CUSIP, TEMP1.VAR, SI. COUNTRY**

**FROM TEMP1**

**INNER JOIN security\_info SI**

**ON TEMP1.CUSIP=SI.CUSIP**

**AND TEMP1.COB\_DATE BETWEEN SI.START\_DATE AND SI.END\_DATE**

**);**

**SELECT CRP.REGION,SUM(TEMP2.VAR) Combined\_VaR\_For\_March**

**FROM TEMP2**

**INNER JOIN country\_region\_map CRP**

**ON TEMP2.COUNTRY=CRP.COUNTRY\_CODE**

**GROUP BY CRP.REGION**

**ORDER BY Combined\_VaR\_For\_March DESC;**

Text, letter

Description automatically generated

**Based on the result above, Africa has the largest combined VaR for the month of March 2018.**

1. What is the weighted average duration for each client for each year? You may use quantity for your weighting. The query should return:

YEAR | FUND\_NAME | WT\_AVG\_DURATION

**Hint:** Weighted average rate is simply the average rate, but we don’t weight the clients equally. For example, if Client A, B, C, D all had a total quantity of 100, then each would weight 25%. Therefor we would do as follows:

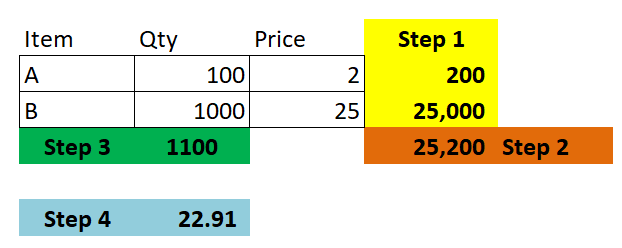
(ClientA\_Duration \* .25 + ClientB\_ Duration \* .25 + ClientC\_ Duration \* .25 + ClientD\_ Duration \* .25)

However, if you look at the SUM of quantity for each day by client, you will see they do not equate to exactly ¼ of the total. So the number .25 will differ for each Client. We get each client’s weight by doing the following:

SUM(ClientA\_Quantity) / SUM(ALLClients\_Quantity)

This will give us the weight to use in the above formula.

For illustration this is a weighted average price diagram. If you plug the base table in Excel you should be able to calculate the weighted average price of 22.91. Once you have done that you now understand the methodology necessary to solve this problem.



**CREATE TEMPORARY TABLE TEMP1(**

**SELECT YEAR(﻿COB\_DATE) Year,FI.FUND\_NAME,RDH.Duration, TDH.QUANTITY**

**from trade\_data\_hist TDH**

**INNER JOIN risk\_data\_hist RDH**

**ON TDH.CUSIP=RDH.CUSIP AND TDH.POSITION\_ID=RDH.POSITION\_ID**

**INNER JOIN fund\_info FI**

**ON TDH.FUND\_ID=FI.FUND\_ID AND TDH.﻿COB\_DATE BETWEEN FI.START\_DATE AND FI.END\_DATE**

**ORDER BY FI.FUND\_NAME, YEAR ASC);**

**CREATE TEMPORARY TABLE TEMP2(**

**SELECT YEAR, FUND\_NAME,(SUM(DURATION\*QUANTITY)/SUM(QUANTITY)) WT\_AVG\_DURATION**

**FROM TEMP1**

**WHERE YEAR=2018**

**GROUP BY FUND\_NAME);**

**SELECT \***

**FROM TEMP2**

**UNION**

**SELECT YEAR, FUND\_NAME, SUM(DURATION\*QUANTITY)/SUM(QUANTITY)**

**FROM TEMP1**

**WHERE YEAR=2019**

**GROUP BY FUND\_NAME;**

Graphical user interface, text

Description automatically generated with medium confidence Graphical user interface, application, table

Description automatically generated with medium confidence

**The result looks like above pictures.**

### **7.2: SUBQUERY**

1. What is a subquery?

**A subquery is a query that is nested inside a SELECT , INSERT , UPDATE , or DELETE statement, or inside another subquery. It writes multiple queries for a single return.**

1. What is more efficient a join or a subquery?

**A join executes faster. The retrieval time of the query using joins almost always will be faster than that of a subquery. By using joins, you can minimize the calculation burden on the database i.e., instead of multiple queries using one join query. This means you can make better use of the database’s abilities to search through, filter, sort, etc.**

1. Provide some code below on when you might use a subquery (should work on our data)

### **SELECT TDH.﻿COB\_DATE, TDH.CUSIP,TDH.FUND\_ID,TDH.QUANTITY,**

### **(SELECT FI.FUND\_NAME**

### **FROM fund\_info FI**

### **WHERE FI.FUND\_ID=TDH.FUND\_ID) FUND\_NAME**

### **FROM trade\_data\_hist TDH**

### **WHERE TDH.FUND\_ID IN(**

### **SELECT FI.FUND\_ID**

### **FROM fund\_info FI**

### **WHERE FI.FUND\_NAME='Whale Rock' ) AND**

### **TDH.﻿COB\_DATE BETWEEN (**

### **SELECT FI.START\_DATE**

### **FROM fund\_info FI**

### **WHERE FI.FUND\_NAME='Whale Rock' ) AND (**

### **SELECT FI.END\_DATE**

### **FROM fund\_info FI**

### **WHERE FI.FUND\_NAME='Whale Rock' );**

Table

Description automatically generated

**The result is like above picture.**

### **7.2: COMMON TABLE EXPRESSION**

1. What is a CTE?

**A Common Table Expression (CTE) is the result set of a query which exists temporarily and for use only within the context of a larger query. Much like a derived table, the result of a CTE is not stored and exists only for the duration of the query.**

1. When might you use a CTE?

**I might use a CTE when I need the following:**

* **Needing to reference a derived table multiple times in a single query**
* **An alternative to creating a view in the database**
* **Performing the same calculation multiple times over across multiple query components**

1. Setup the following query using CTEs:
2. Create a temporary table that has three columns: Fund\_Name, Date, Quantity
3. Create a second table that has three columns: Fund \_Name, Date, VaR99
4. Join both CTEs to provide a single result of Client, Date, Quantity, and VaR99

**WITH TEMP1 AS (SELECT FI.FUND\_NAME,TDH.COB\_DATE, SUM(TDH.QUANTITY) QTY**

**FROM trade\_data\_hist TDH**

**INNER JOIN fund\_info FI**

**ON TDH.FUND\_ID=FI.FUND\_ID AND TDH.COB\_DATE BETWEEN FI.START\_DATE AND FI.END\_DATE**

**GROUP BY FI.FUND\_NAME,TDH.COB\_DATE),**

**TEMP2 AS (SELECT FI.FUND\_NAME,TDH.COB\_DATE, RDH.VaR VaR99**

**FROM trade\_data\_hist TDH**

**INNER JOIN fund\_info FI**

**ON TDH.FUND\_ID=FI.FUND\_ID AND TDH.COB\_DATE BETWEEN FI.START\_DATE AND FI.END\_DATE**

**INNER JOIN risk\_data\_hist RDH**

**ON TDH.POSITION\_ID=RDH.POSITION\_ID)**

**SELECT**

**TEMP1.FUND\_NAME, TEMP1.COB\_DATE,TEMP1.QTY, TEMP2.VaR99**

**FROM TEMP1**

**INNER JOIN TEMP2**

**WHERE TEMP1.COB\_DATE=TEMP2.COB\_DATE AND TEMP1.FUND\_NAME=TEMP2.FUND\_NAME;**

**Table

Description automatically generated with medium confidence**

**The result looks like above picture.**

### **7.3: DETERMINISTIC FUNCTION**

1. What is a deterministic function?

**It allows for a deterministic output to a controlled set of inputs. Deterministic functions always result in the same output every time they are called with a fixed set of input values and given the same condition of the database.**

1. Create a function that accepts a date and a string parameter: “Y, Q, M, D”. Based on the parameter selected, the function should return either the year, quarter (Q1, Q2, Q3, Q4), month, or day of month.

**CREATE DEFINER=`root`@`localhost` FUNCTION `new\_function`(DATE\_PARAMETER INT,STRING\_PAPAMETER TEXT) RETURNS text CHARSET utf8mb4**

**DETERMINISTIC**

**BEGIN**

**RETURN ( CASE WHEN STRING\_PAPAMETER='Y' THEN YEAR(DATE\_PARAMETER)**

**WHEN STRING\_PAPAMETER='M' THEN MONTH(DATE\_PARAMETER)**

**WHEN STRING\_PAPAMETER='D' THEN DAY(DATE\_PARAMETER)**

**WHEN STRING\_PAPAMETER='Q' AND MONTH(DATE\_PARAMETER) BETWEEN 1 AND 3 THEN 'Q1'**

**WHEN STRING\_PAPAMETER='Q' AND MONTH(DATE\_PARAMETER) BETWEEN 4 AND 6 THEN 'Q2'**

**WHEN STRING\_PAPAMETER='Q' AND MONTH(DATE\_PARAMETER) BETWEEN 7 AND 9 THEN 'Q3'**

**WHEN STRING\_PAPAMETER='Q' AND MONTH(DATE\_PARAMETER) BETWEEN 10 AND 12 THEN 'Q4'**

**END**

**) ;**

**END**

**SELECT**

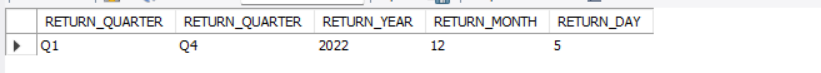
**DBO.new\_function(20230105,'Q') RETURN\_QUARTER,**

**DBO.new\_function(20221205,'Q') RETURN\_QUARTER,**

**DBO.new\_function(20221205,'Y') RETURN\_YEAR,**

**DBO.new\_function(20221205,'M') RETURN\_MONTH,**

**DBO.new\_function(20221205,'D') RETURN\_DAY**



**Result should look live above picture.**

1. Create a function that accepts a cusip and a date and returns the TOTAL QUANTITY, TOTAL MARKET\_VALUE, AVG VAR, and AVG DURATION of that cusip.

**CREATE DEFINER=`root`@`localhost` FUNCTION `new\_function2`(cu\_sip TEXT, COB INT) RETURNS text CHARSET utf8mb4**

**DETERMINISTIC**

**BEGIN**

**RETURN (WITH TEMP1 AS (SELECT POSITION\_ID, SUM(VaR) VAR, SUM(Duration) DURATION FROM risk\_data\_hist GROUP BY POSITION\_ID)**

**SELECT CONCAT('CUSIP:',TDH.CUSIP, ', TOTAL\_QUANTITY:', SUM(TDH.QUANTITY), ' , Total\_Market\_Value:',**

**SUM(TDH.QUANTITY\*SPH.CLOSE\_PRICE\_USD), ', AVG\_VAR:',**

**AVG(TEMP1.VAR), ' , AVG\_DURATION:',**

**AVG(TEMP1.DURATION)) RESULT**

**FROM trade\_data\_hist TDH**

**INNER JOIN security\_price\_hist SPH**

**ON TDH.COB\_DATE=SPH.COB\_DATE AND TDH.CUSIP=SPH.CUSIP**

**INNER JOIN TEMP1**

**ON TDH.POSITION\_ID=TEMP1.POSITION\_ID**

**WHERE TDH.CUSIP=cu\_sip AND TDH.COB\_DATE=COB**

**GROUP BY TDH.CUSIP);**

**END**

**SELECT DBO.new\_function2('C207240619',20180102) RESULT**



**The result looks like above picture.**

### **7.4: TEMPORARY TABLE**

1. What is a temporary table?

**A temporary table is a base table that is not stored in the database but instead exists only while the database session in which it was created is active. It allows for a static table which has a predetermined dataset.**

1. Create a temporary table with two columns: CUSIP and REGION.

**CREATE TEMPORARY TABLE CUSIP\_REGOIN(**

**SELECT TDH.CUSIP, FI.REGION**

**FROM trade\_data\_hist TDH**

**INNER JOIN fund\_info FI**

**ON TDH.FUND\_ID=FI.FUND\_ID AND TDH.COB\_DATE BETWEEN FI.START\_DATE AND FI.END\_DATE**

**WHERE CUSIP<>''**

**)**

1. Load the table with all active cusips as of Jan 2 2018 and their respective regions.

**CREATE TEMPORARY TABLE CUSIP\_REGOIN1(**

**SELECT TDH.COB\_DATE,TDH.CUSIP, FI.REGION**

**FROM trade\_data\_hist TDH**

**INNER JOIN fund\_info FI**

**ON TDH.FUND\_ID=FI.FUND\_ID AND TDH.COB\_DATE BETWEEN FI.START\_DATE AND FI.END\_DATE**

**WHERE CUSIP<>''**

**);**

**SELECT CUSIP, REGION**

**FROM CUSIP\_REGOIN1**

**WHERE COB\_DATE=20180102**

1. Join the above temporary table with RISK\_DATA\_HIST to retrieve the SUM of VaR for each date for each region.

**SELECT CUSIP\_REGOIN1.\*, SUM(RDH.VaR) VAR**

**FROM CUSIP\_REGOIN1**

**INNER JOIN risk\_data\_hist RDH**

**ON CUSIP\_REGOIN1.COB\_DATE=RDH.COB\_DATE AND CUSIP\_REGOIN1.CUSIP=RDH.CUSIP**

**GROUP BY CUSIP\_REGOIN1.COB\_DATE, CUSIP\_REGOIN1.REGION;**

**Graphical user interface, text, application

Description automatically generated**

**The result looks like above picture.**

### **7.5: STORED PROCEDURES**

1. Please write your own useful stored procedure on our data.

**CREATE DEFINER=`root`@`localhost` PROCEDURE `SP\_QUANTITY\_BY\_DATE`(START\_DATE INT, END\_DATE INT)**

**BEGIN**

**SELECT COB\_DATE, POSITION\_ID, QUANTITY**

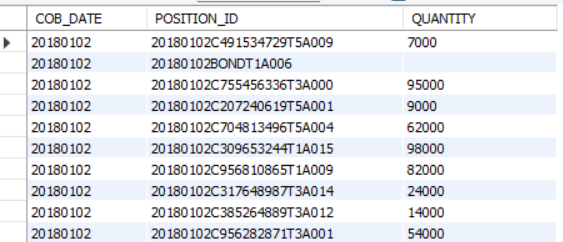
**FROM TRADE\_DATA\_HIST**

**WHERE COB\_DATE BETWEEN START\_DATE AND END\_DATE**

**GROUP BY COB\_DATE, POSITION\_ID;**

**END**

**CALL DBO.SP\_QUANTITY\_BY\_DATE(20180102, 20190310);**

****

**The result looks like above picture.**

1. Write a stored procedure that accepts a start date, end date, and a fund name. The SP should return: Quarter, Market Value, and % Market Value change over previous quarter.

CREATE DEFINER=`root`@`localhost` PROCEDURE `SP\_Q7`(START\_DATE INT, END\_DATE INT, FUND\_NAME TEXT)

**BEGIN**

**WITH TEMP1 AS(**

**SELECT TDH.COB\_DATE, TDH.CUSIP, TDH.FUND\_ID, FI.FUND\_NAME, SUM(TDH.QUANTITY\*SPH.CLOSE\_PRICE\_USD) MARKET\_VALUE**

**FROM trade\_data\_hist TDH**

**INNER JOIN security\_price\_hist SPH**

**ON TDH.COB\_DATE=SPH.COB\_DATE AND TDH.CUSIP=SPH.CUSIP**

**INNER JOIN fund\_info FI**

**ON TDH.FUND\_ID=FI.FUND\_ID**

**WHERE TDH.COB\_DATE BETWEEN FI.START\_DATE AND FI.END\_DATE**

**GROUP BY TDH.COB\_DATE, TDH.CUSIP),**

**TEMP2 AS(SELECT YEAR(T1.COB\_DATE) YEAR, QUARTER(T1.COB\_DATE) QUARTER,**

**T1.FUND\_NAME, SUM(T1.MARKET\_VALUE) MARKET\_VALUE, (SUM(T2.MARKET\_VALUE)/SUM(T1.MARKET\_VALUE))-1 Market\_Value\_Change**

**FROM TEMP1 T1**

**LEFT JOIN TEMP1 T2**

**ON T1.FUND\_NAME=T2.FUND\_NAME AND T1.COB\_DATE=DATE\_SUB(T2.COB\_DATE, INTERVAL 1 QUARTER)**

**WHERE T1.FUND\_NAME=FUND\_NAME AND T1.COB\_DATE BETWEEN START\_DATE AND END\_DATE**

**GROUP BY YEAR,QUARTER)**

**SELECT QUARTER, MARKET\_VALUE, Market\_Value\_Change**

**FROM TEMP2;**

**END**

**CALL DBO.SP\_Q7(20180101, 20190401, 'Whale Rock')**

**Graphical user interface, text, application

Description automatically generated**

**The result looks like above picture.**

**Comments:**

**Grade: 88/100**

**7.1.1 Very much overkill. You can just write a simple query with an inner join on FUND and on PRICE and can easily return the result.**

**7.1.2 Same, complete overkill. This can be done in a single query.**

**7.1.3 Same issue.**

**7.4.2 Just asked to create a table. It did not ask for data to be loaded.**

**7.4.3 Similar issue. This was just for an insert command not to create a table.**