NOTE: I repeated the lab and still got the same results as I got before. According to order quantity, chocolate cake is the highest selling product with quantity 30 and Coffee cake is the lowest selling product with order quantity 19.

Department of Software Engineering Mehran University of Engineering and Technology, Jamshoro			
Course: SWE324 - Data Warehousing and Data Mining			
Instructor	Rabeea Jaffari	**Practical/Lab No.**	04
Date		**CLOs**	CLO-4: P3 & P4
Signature		**Assessment Score**	1 Mark

Topic	**To familiar with OLTP system reporting**
Objectives	- To learn report generation in OLTP systems

Lab Discussion: Theoretical concepts and Procedural steps

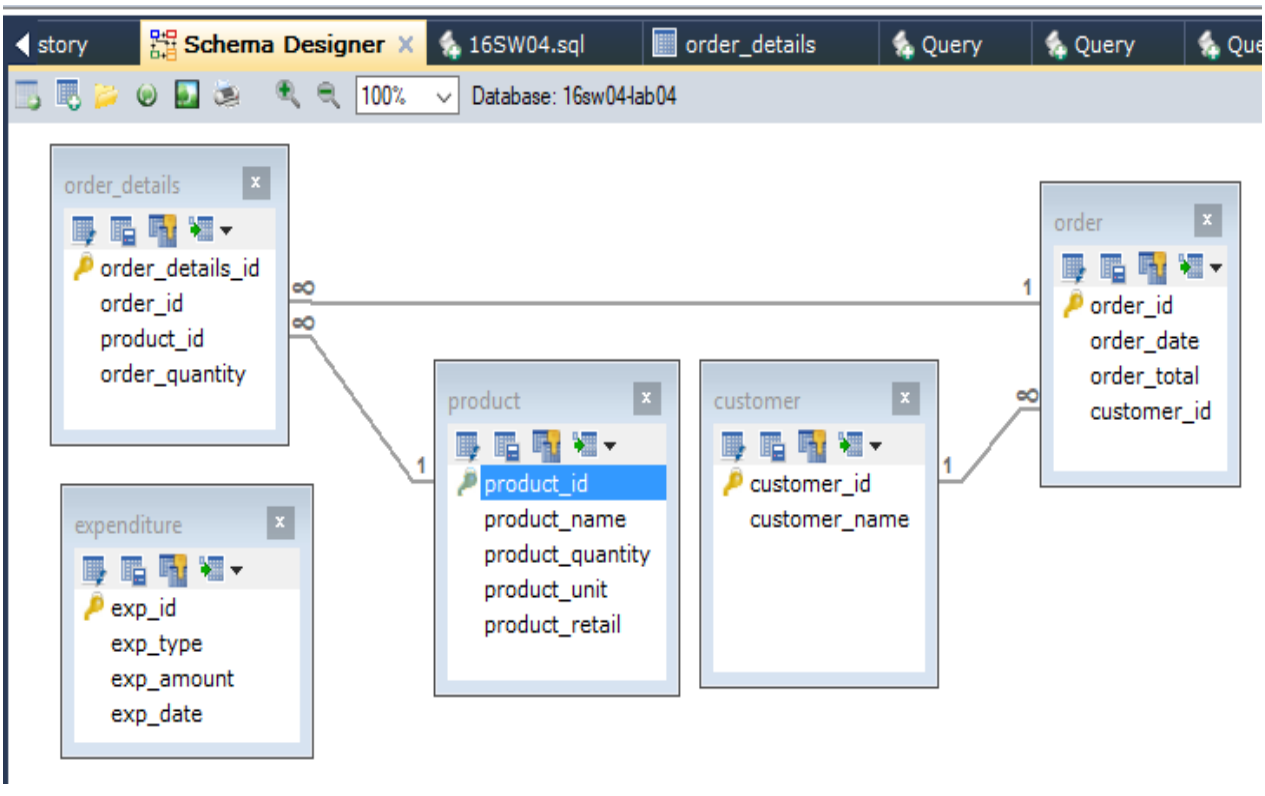
Lab Tasks

Submission Date: 23-04-19

Generate an OLTP system report which displays the following results:

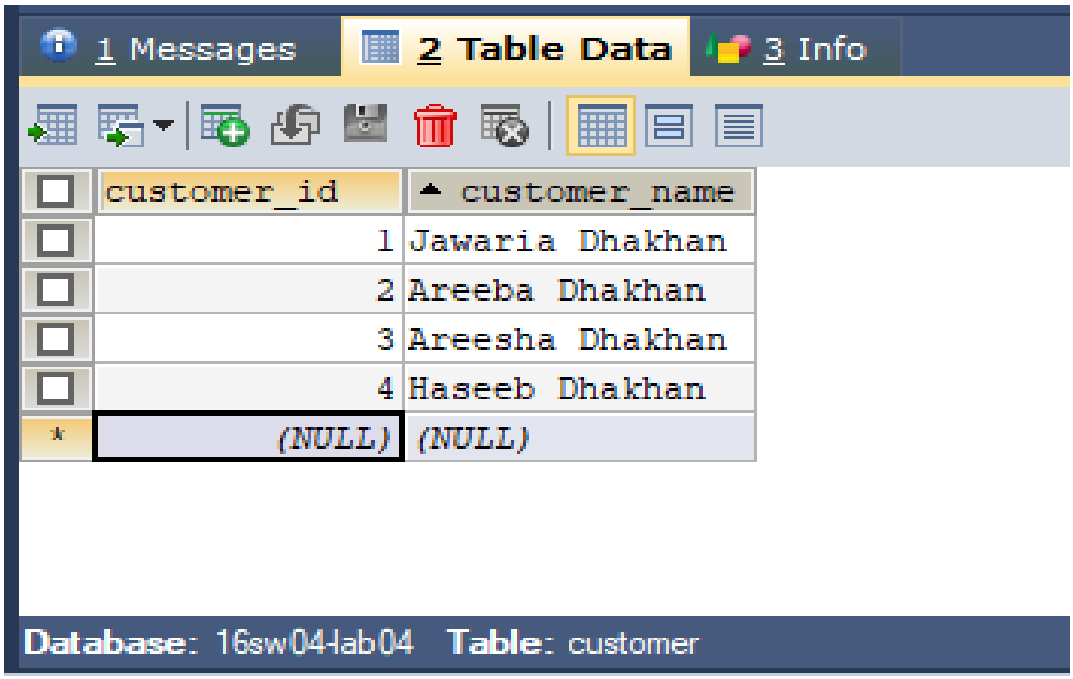
1. **Total sales in a month** (Use only order table).
2. **Profit/loss in a month** (Use product (to account for purchasing costs), expenditure as well as order tables).
3. **Highest selling product of the month**(By highest sold quantity)
4. **Lowest selling product of the month**(By lowest sold quantity)

Hint: Create a report table to hold all the above results from queries after they are executed in the stored procedure and then create a stored procedure in the similar manner as shown above.

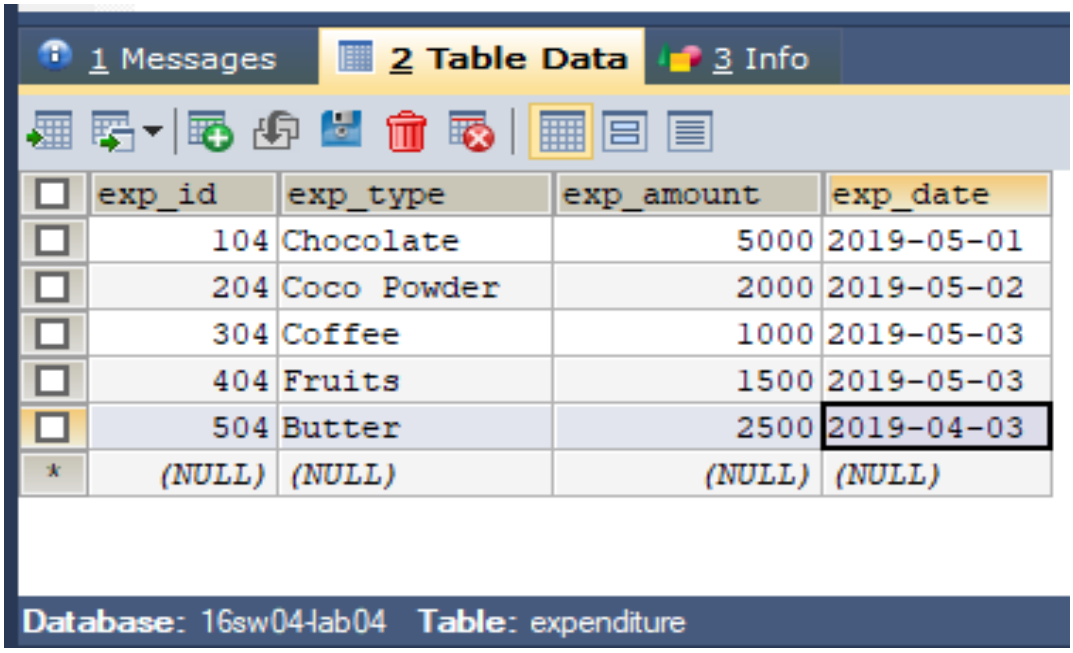


1. **Tables in SQLYog**

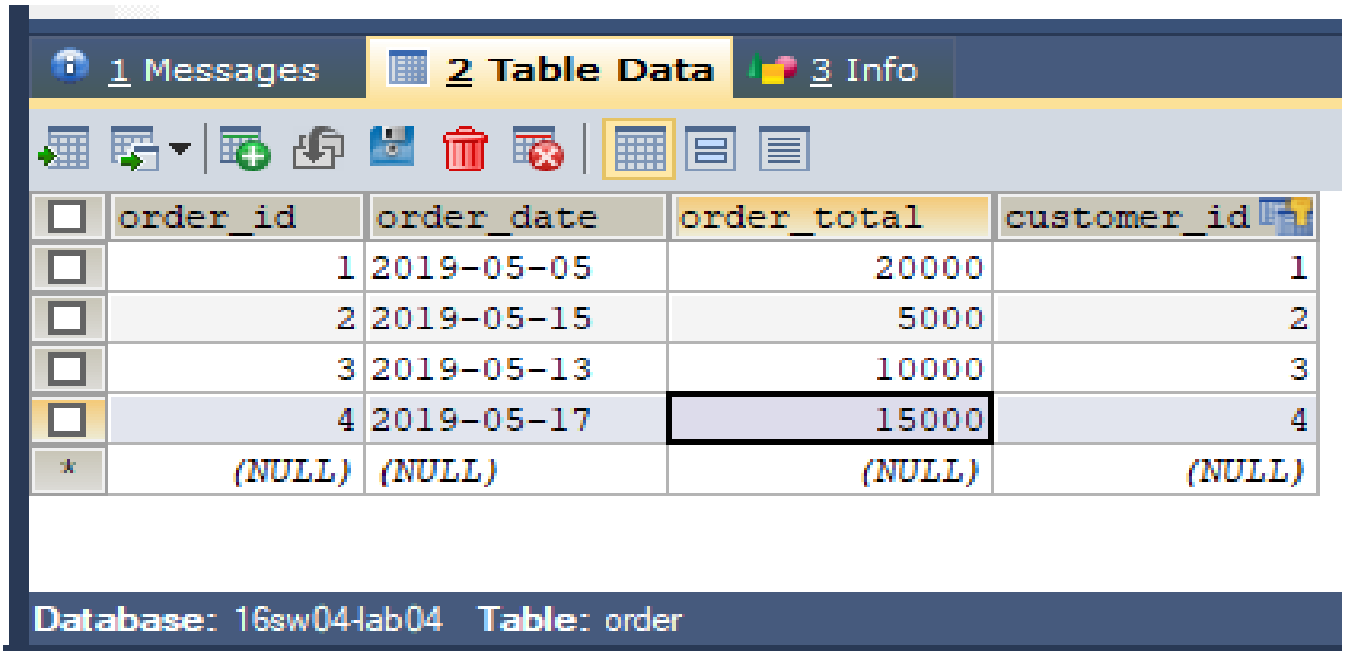
Customer Table



Expenditure Table



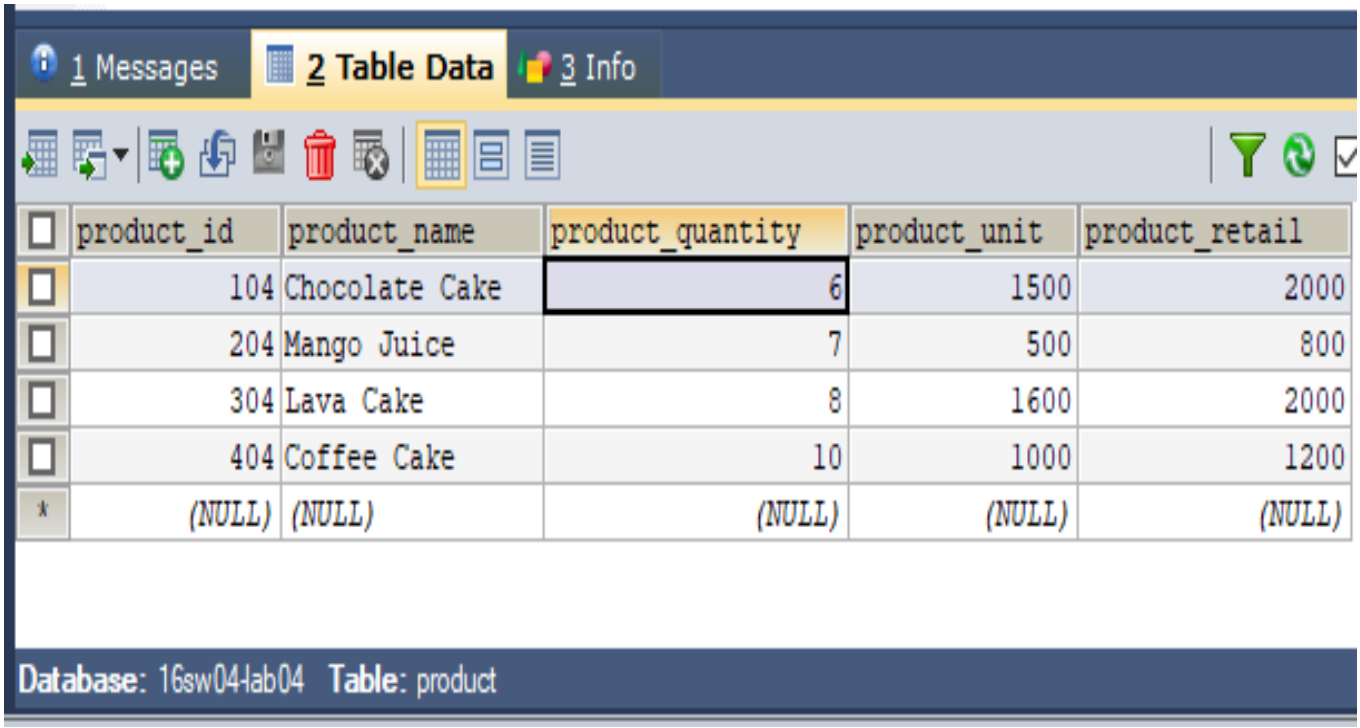
Order Table


1 Messages | 2 Table Data | 3 Info

order_id	order_date	order_total	customer_id
1	2019-05-05	20000	1
2	2019-05-15	5000	2
3	2019-05-13	10000	3
4	2019-05-17	15000	4
(NULL)	*(NULL)*	*(NULL)*	*(NULL)*

Database: 16sw04lab04 Table: order

Product Table


1 Messages | 2 Table Data | 3 Info

product_id	product_name	product_quantity	product_unit	product_retail
104	Chocolate Cake	6	1500	2000
204	Mango Juice	7	500	800
304	Lava Cake	8	1600	2000
404	Coffee Cake	10	1000	1200
(NULL)	*(NULL)*	*(NULL)*	*(NULL)*	*(NULL)*

Database: 16sw04lab04 Table: product

Order_details Table

1 Messages | 2 Table Data | 3 Info

order_details_id	order_id	product_id	order_quantity
1	1	104	30
2	2	204	20
3	3	304	25
4	4	404	19
(NULL)	*(NULL)*	*(NULL)*	*(NULL)*

Database: 16sw04-lab04 Table: order_details

2. **Create a report table to hold all the results from queries.**

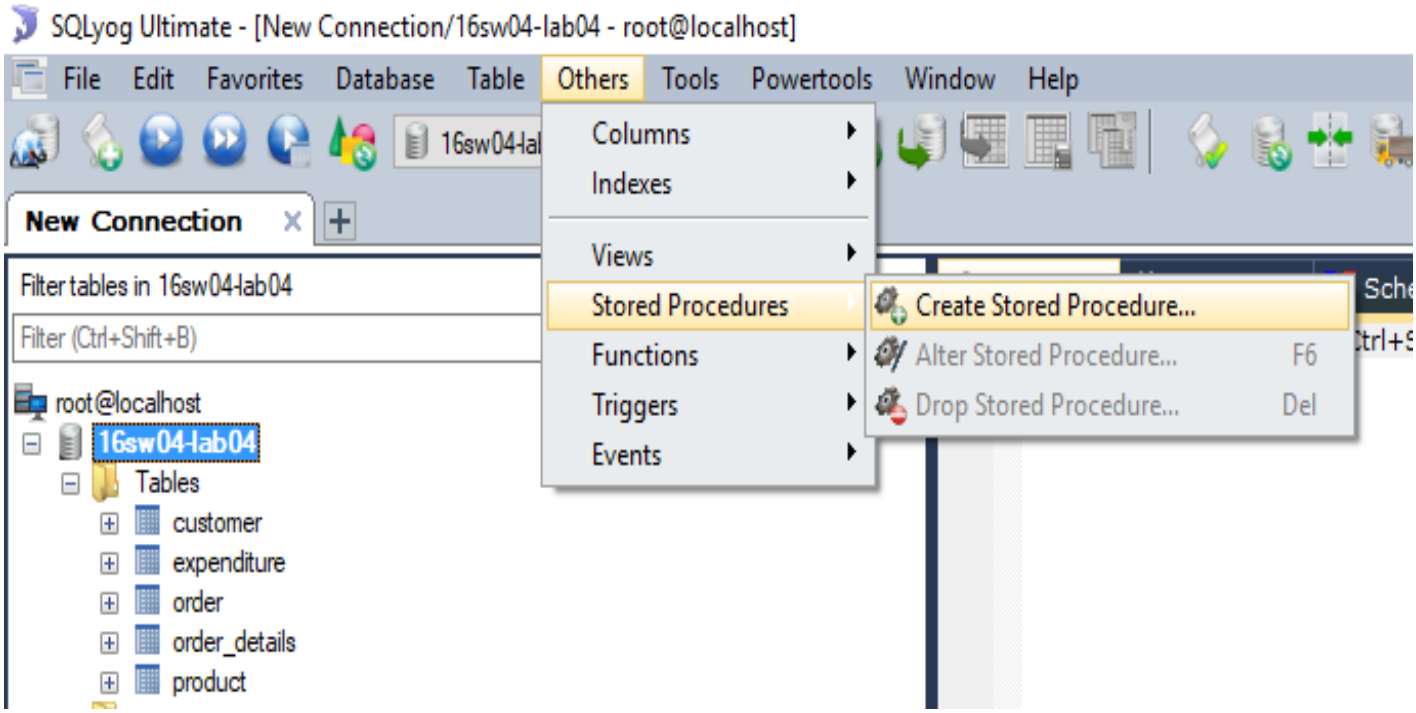
Query | History | Schema Designer | 16SW04.sql* | Query | product | Query | report

Table Name: report
Engine: InnoDB
Database: 16sw04-lab04
Character Set: latin1
Collation: latin1_swedish_ci

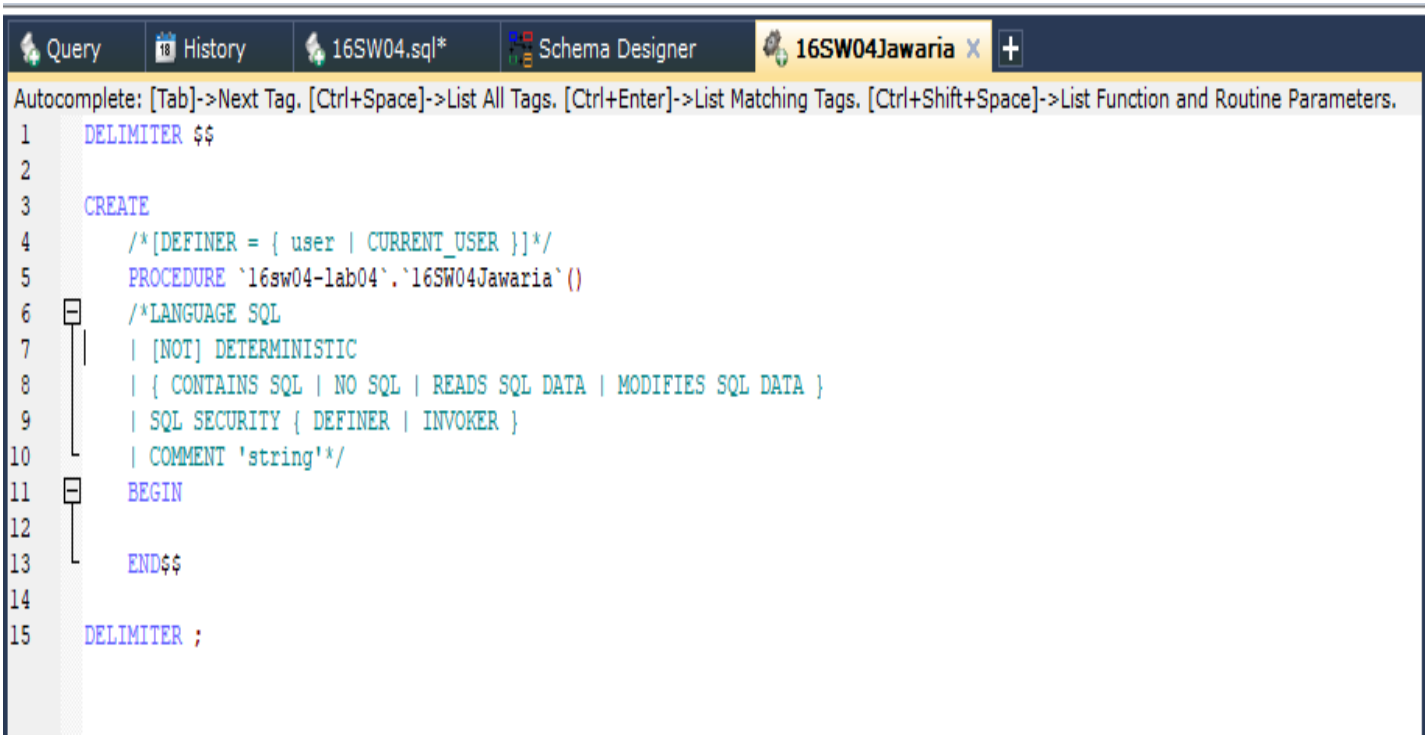
1 Columns | 2 Indexes | 3 Foreign Keys | 4 Advanced | 5 SQL Preview

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
Query	varchar	255							
Result	varchar	255							

3. Create Stored Procedure



4. Create a Stored Procedure named as '16SW04-Jawaria'



5. **Write following queries within the "BEGIN" and "END" section of the stored procedure:**

1. **Total sales in a month** (Use only order table).
2. **Profit/loss in a month** (Use product (to account for purchasing costs), expenditure as well as order tables).
3. **Highest selling product of the month**(By highest sold quantity)
4. **Lowest selling product of the month**(By lowest sold quantity)

Query | History | Schema Designer | 16SW04.sql* | Query | product | Query | report | Query

Autocomplete: [Tab]->Next Tag. [Ctrl+Space]->List All Tags. [Ctrl+Enter]->List Matching Tags. [Ctrl+Shift+Space]->List Function and Routine Parameters.

```
1   DELIMITER $$
2   DROP PROCEDURE IF EXISTS `16SW04Jawaria`$$  CREATE
3       /*[DEFINER = { user | CURRENT_USER }]*/
4       PROCEDURE `16sw04-lab04`.`16SW04Jawaria`()
5       /*LANGUAGE SQL
6       | [NOT] DETERMINISTIC
7       | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
8       | SQL SECURITY { DEFINER | INVOKER }
9       | COMMENT 'string'*/
10      BEGIN
11  SELECT @productexp:=SUM(product_unit*product_quantity) FROM product;
12  SELECT @expenditure:=SUM(exp_amount) FROM expenditure WHERE exp_date BETWEEN '2019-05-01' AND '2019-05-18';
13  SELECT @totalExp:=SUM(@productexp+@expenditure);
14
15  /*1. Total sales in a month (Use only order table)16SW04 */
16  SELECT @totalsales:=SUM(order_total) FROM `order` WHERE `order_date` LIKE '2019-05%';
17  INSERT INTO report(QUERY, Result) VALUES ('TOTAL SALES',@totalsales);
18
19  /*2. Profit/loss in a month (Use product (to account for purchasing costs), expenditure as well as order tables). 16SW04 */
20  SELECT @profit:=@totalsales-@totalExp;
21  INSERT INTO report(QUERY, Result) VALUES ('PROFIT/LOSS',@profit);
22
23  /*3. Highest selling product of the month(By highest sold quantity)16SW04 */
24  SELECT @high_sale_product:=product.product_name FROM product product INNER JOIN order_details order_details ON
25  product.product_id=(SELECT order_details.product_id WHERE order_details.order_quantity=
26  (SELECT  MAX(order_details.order_quantity) FROM order_details order_details));
27  INSERT INTO report(QUERY, Result) VALUES ('HIGHEST SELLING PRODUCT',@high_sale_product);
28
29  /*4. Lowest selling product of the month(By lowest sold quantity) 16SW04*/
30   SELECT @low_sale_product:=product.product_name FROM product product INNER JOIN order_details order_details ON
31  product.product_id=(SELECT order_details.product_id WHERE order_details.order_quantity=
32  (SELECT  MIN(order_details.order_quantity)
33  FROM order_details order_details));
34  INSERT INTO report(QUERY, Result) VALUES ('LOWEST SELLING PRODUCT',@low_sale_product);
35  SELECT * FROM report;
36     END$$  DELIMITER ;
```

6. **Execute the procedure**

Query | History | Schema Designer | 16SW04.sql* | Query | product | Query | report

Autocomplete: [Tab]->Next Tag. [Ctrl+Space]->List All Tags. [Ctrl+Enter]->List Matching Tags. [Ctrl+Shift+Space]->List Function and Routine Par

```
1   DELIMITER $$
2   DROP PROCEDURE IF EXISTS `16SW04Jawaria`$$
3   CREATE
4       /*[DEFINER = { user | CURRENT_USER }]*/
5       PROCEDURE `16sw04-lab04`.`16SW04Jawaria`()
6       /*LANGUAGE SQL
7       | [NOT] DETERMINISTIC
8       | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
9       | SQL SECURITY { DEFINER | INVOKER }
10      | COMMENT 'string'*/
11      BEGIN
12  SELECT @productexp:=SUM(product_unit*product_quantity) FROM product;
13  SELECT @expenditure:=SUM(exp_amount) FROM expenditure WHERE exp_date BETWEEN '2019-05-01' AND '2019-05-18';
14  SELECT @totalExp:=SUM(@productexp+@expenditure);
15
16  /*1. Total sales in a month (Use only order table)16SW04 */
17  SELECT @totalsales:=SUM(order_total) FROM `order` WHERE `order_date` LIKE '2019-05%';
18  INSERT INTO report(QUERY, Result) VALUES ('TOTAL SALES',@totalsales);
19
```

1 Messages | 2 Table Data | 3 Info

```
2 queries executed, 2 success, 0 errors, 0 warnings

Query: drop procedure if exists `16SW04Jawaria`

0 row(s) affected

Execution Time : 0.011 sec
Transfer Time  : 1.576 sec
Total Time     : 1.587 sec
------------------------------------------------

Query: CREATE PROCEDURE `16sw04-lab04`.`16SW04Jawaria`() BEGIN SELECT @productexp:=SUM(product_unit*produ
SEL...
```

All

7. **Write "CALL 16SW04Jawaria()" syntax to execute all the statements together.**