**A MINI PROJECT**

**ON**

**LIBRARY MANAGEMENT DATABASE**

**(USING ORACAL SQL)**

**(25/01/2023 – 05/02/2023)**

**SUBMITTED TO: SUBMITTED BY:**

**SKILL ACADEMY KISHAN KUMAR**

**(BUSSINESS ANALYST TRAINEE)**

**INDEX**

**Title Page no.**

1. Introduction 3

1.1)Problem Statement

1.2)Objective of the study

1. Database Schema 4
2. Database Creation (script) 5
3. Application of Database(cases) 10
4. Representation (Tables) 14
5. Result 26
6. Conclusion 26

1. **Introduction:-**

This presents implementation of cost effective Library Management Database with the use of SQL. This framework is intended to help and give help to satisfy the needs

of Library management.

* 1. **Problem Statement**

In the present day library management is becoming essential for the propose of improving services. There is complexity , not cost effective and easy of configure . Convenience and ease of using library what is offering library management database.

* 1. **Objective of Study**
* To develop a library management database by using oracle SQL
* To design a user friendly and a safe management database

1. **Database Schema:-**

A schema is a collection of database objects. Schema objects are logical structures created by users. Objects such as tables or indexes hold data.

**LMS\_MEMBERS**

**LMS\_SUPPLIER\_DETAILS**

**LMS\_BOOK\_ DETAILS**

**LMS\_FINE\_ DETAILS**

**LMS\_BOOK\_ ISSUE**

1. **Database Creation:-**

**Table LMS MEMBERS**

CREATE TABLE LMS\_MEMBERS

(

MEMBER\_ID VARCHAR2(10 BYTE), // COLUMNS CREATED

MEMBER\_NAME VARCHAR2(30 BYTE) NOT NULL ,

CITY VARCHAR2(20 BYTE),

DATE\_REGISTER DATE NOT NULL ,

DATE\_EXPIRE DATE,

MEMBERSHIP\_STATUS VARCHAR2(15 BYTE) NOT NULL ,

CONSTRAINT PK\_MBR\_ID PRIMARY KEY (MEMBER\_ID)

)

// INSERT VALUE

Insert into LMS\_MEMBERS (MEMBER\_ID,MEMBER\_NAME,CITY,DATE\_REGISTER,DATE\_EXPIRE,MEMBERSHIP\_STATUS) values ('LM001','Akshay','CHENNAI',to\_date('12-06-18','DD-MM-RR'),to\_date('26-10-20','DD-MM-RR'),'Temporary')

Insert into LMS\_MEMBERS (MEMBER\_ID,MEMBER\_NAME,CITY,DATE\_REGISTER,DATE\_EXPIRE,MEMBERSHIP\_STATUS) values ('LM003','Ashish','CHENNAI',to\_date('12-06-18','DD-MM-RR'),to\_date('17-08-20','DD-MM-RR'),'Permanent')

Insert into LMS\_MEMBERS (MEMBER\_ID,MEMBER\_NAME,CITY,DATE\_REGISTER,DATE\_EXPIRE,MEMBERSHIP\_STATUS) values ('LM002','Amruta','PUNE',to\_date('02-03-20','DD-MM-RR'),to\_date('03-06-20','DD-MM-RR'),'Temporary')

Insert into LMS\_MEMBERS (MEMBER\_ID,MEMBER\_NAME,CITY,DATE\_REGISTER,DATE\_EXPIRE,MEMBERSHIP\_STATUS) values ('LM005','Gautam','BANGALORE',to\_date('12-06-18','DD-MM-RR'),to\_date('06-11-20','DD-MM-RR'),'Temporary')

Insert into LMS\_MEMBERS (MEMBER\_ID,MEMBER\_NAME,CITY,DATE\_REGISTER,DATE\_EXPIRE,MEMBERSHIP\_STATUS) values ('LM004','Bhakti','CHENNAI',to\_date('12-06-18','DD-MM-RR'),to\_date('12-05-20','DD-MM-RR'),'Temporary')

**Table LMS SUPPLIERS DETAILS**

CREATE TABLE LMS\_SUPPLIERS\_DETAILS

(

SUPPLIER\_ID VARCHAR2(3 BYTE),

SUPPLIER\_NAME VARCHAR2(30 BYTE) NOT NULL ,

ADDRESS VARCHAR2(50 BYTE),

CONTACT NUMBER(10,0) NOT NULL ,

EMAIL VARCHAR2(15 BYTE) NOT NULL ,

CONSTRAINT PK\_SP\_ID PRIMARY KEY (SUPPLIER\_ID)

)

Insert into LMS\_SUPPLIERS\_DETAILS (SUPPLIER\_ID,SUPPLIER\_NAME,ADDRESS,CONTACT,EMAIL) values ('S01','SINGAPORE SHOPPEE','CHENNAI',9894123555,'sing@gmail.com')

Insert into LMS\_SUPPLIERS\_DETAILS (SUPPLIER\_ID,SUPPLIER\_NAME,ADDRESS,CONTACT,EMAIL) values ('S02','JK Stores','MUMBAI',9940123450,'jks@yahoo.com')

Insert into LMS\_SUPPLIERS\_DETAILS (SUPPLIER\_ID,SUPPLIER\_NAME,ADDRESS,CONTACT,EMAIL) values ('S03','ROSE BOOK STORE','TRIVANDRUM',9444411222,'rose@gmail.com')

Insert into LMS\_SUPPLIERS\_DETAILS (SUPPLIER\_ID,SUPPLIER\_NAME,ADDRESS,CONTACT,EMAIL) values ('S04','KAVARI STORE','DELHI',8630001452,'kavi@redif.com')

**Table LMS BOOK DETAILS**

CREATE TABLE LMS\_BOOK\_DETAILS

(

BOOK\_CODE VARCHAR2(10 BYTE),

BOOK\_TITLE VARCHAR2(100 BYTE) NOT NULL ,

CATEGORY VARCHAR2(15 BYTE) NOT NULL ,

AUTHOR VARCHAR2(30 BYTE) NOT NULL ,

PUBLICATION VARCHAR2(30 BYTE),

PUBLISH\_DATE DATE,

BOOK\_EDITION NUMBER,

PRICE NUMBER(8,2) NOT NULL ,

RACK\_NUM VARCHAR2(3 BYTE),

DATE\_ARRIVAL DATE,

SUPPLIER\_ID VARCHAR2(3 BYTE),

CONSTRAINT PK\_BOOK\_CODE PRIMARY KEY (BOOK\_CODE),

CONSTRAINT FK\_SP\_ID FOREIGN KEY (SUPPLIER\_ID) REFERENCES LMS\_SUPPLIERS\_DETAILS (SUPPLIER\_ID)

)

Insert into LMS\_BOOK\_DETAILS (BOOK\_CODE,BOOK\_TITLE,CATEGORY,AUTHOR,PUBLICATION,PUBLISH\_DATE,BOOK\_EDITION,PRICE,RACK\_NUM,DATE\_ARRIVAL,SUPPLIER\_ID) values ('BL000001','Java How To Do Program','JAVA','Paul J. Deitel','Prentice Hall',to\_date('10-12-99','DD-MM-RR'),6,600,'A1',to\_date('10-05-11','DD-MM-RR'),'S01')

Insert into LMS\_BOOK\_DETAILS (BOOK\_CODE,BOOK\_TITLE,CATEGORY,AUTHOR,PUBLICATION,PUBLISH\_DATE,BOOK\_EDITION,PRICE,RACK\_NUM,DATE\_ARRIVAL,SUPPLIER\_ID) values ('BL000002','Java: The Complete Reference ','JAVA','Herbert Schildt','Tata Mcgraw Hill ',to\_date('10-10-11','DD-MM-RR'),5,750,'A1',to\_date('10-05-11','DD-MM-RR'),'S03')

Insert into LMS\_BOOK\_DETAILS (BOOK\_CODE,BOOK\_TITLE,CATEGORY,AUTHOR,PUBLICATION,PUBLISH\_DATE,BOOK\_EDITION,PRICE,RACK\_NUM,DATE\_ARRIVAL,SUPPLIER\_ID) values ('BL000007','Let Us C','C','Yashavant Kanetkar ','BPB Publications',to\_date('11-12-10','DD-MM-RR'),9,500,'A3',to\_date('03-11-10','DD-MM-RR'),'S03')

Insert into LMS\_BOOK\_DETAILS (BOOK\_CODE,BOOK\_TITLE,CATEGORY,AUTHOR,PUBLICATION,PUBLISH\_DATE,BOOK\_EDITION,PRICE,RACK\_NUM,DATE\_ARRIVAL,SUPPLIER\_ID) values ('BL000008','Python Cookbook: Recipes for Mastering Python 3','Python','Brian K. Jones','O''Reilly Media',to\_date('10-05-13','DD-MM-RR'),3,350,null,to\_date('10-06-19','DD-MM-RR'),'S02')

Insert into LMS\_BOOK\_DETAILS (BOOK\_CODE,BOOK\_TITLE,CATEGORY,AUTHOR,PUBLICATION,PUBLISH\_DATE,BOOK\_EDITION,PRICE,RACK\_NUM,DATE\_ARRIVAL,SUPPLIER\_ID) values ('BL000012','C# 8.0 Pocket Reference: Instant Help for C# 8.0 Programmers','C#','Joseph Albahari ','O''Reilly Media',to\_date('10-12-19','DD-MM-RR'),1,890,null,null,'S03')

Insert into LMS\_BOOK\_DETAILS (BOOK\_CODE,BOOK\_TITLE,CATEGORY,AUTHOR,PUBLICATION,PUBLISH\_DATE,BOOK\_EDITION,PRICE,RACK\_NUM,DATE\_ARRIVAL,SUPPLIER\_ID) values ('BL000011','C# 8.0 and .NET Core 3.0 – Modern Cross-Platform Development','C#','Mark J. Price','Packt Publishing',to\_date('31-10-19','DD-MM-RR'),4,410,'A3',to\_date('10-06-19','DD-MM-RR'),null)

**Table LMS FINE DETAILS**

CREATE TABLE LMS\_FINE\_DETAILS

(

FINE\_RANGE VARCHAR2(3 BYTE),

FINE\_AMOUNT NUMBER(10,2) NOT NULL ,

MAX\_DAYS\_DELAYED NUMBER,

CONSTRAINT PK\_FR PRIMARY KEY (FINE\_RANGE)

)

Insert into LMS\_FINE\_DETAILS (FINE\_RANGE,FINE\_AMOUNT,MAX\_DAYS\_DELAYED) values ('R1',20,5)

Insert into LMS\_FINE\_DETAILS (FINE\_RANGE,FINE\_AMOUNT,MAX\_DAYS\_DELAYED) values ('R2',50,10)

Insert into LMS\_FINE\_DETAILS (FINE\_RANGE,FINE\_AMOUNT,MAX\_DAYS\_DELAYED) values ('R3',75,15)

Insert into LMS\_FINE\_DETAILS (FINE\_RANGE,FINE\_AMOUNT,MAX\_DAYS\_DELAYED) values ('R4',100,20)

**Table LMS BOOK ISSUE**

CREATE TABLE LMS\_BOOK\_ISSUE

(

BOOK\_ISSUE\_NO NUMBER,

MEMBER\_ID VARCHAR2(10 BYTE) NOT NULL ,

BOOK\_CODE VARCHAR2(10 BYTE) NOT NULL ,

DATE\_ISSUE DATE NOT NULL ,

DATE\_RETURN DATE NOT NULL ,

DATE\_RETURNED DATE ,

FINE\_RANGE VARCHAR2(3 BYTE),

CONSTRAINT PK\_BOOK\_ISSUE\_NO PRIMARY KEY (BOOK\_ISSUE\_NO),

CONSTRAINT FK\_MEM\_ID FOREIGN KEY (MEMBER\_ID) REFERENCES LMS\_MEMBERS (MEMBER\_ID) ,

CONSTRAINT FK\_BOOKDETAIL FOREIGN KEY (BOOK\_CODE) REFERENCES LMS\_BOOK\_DETAILS (BOOK\_CODE) ,

CONSTRAINT FK\_FINEDETAIL FOREIGN KEY (FINE\_RANGE) REFERENCES LMS\_FINE\_DETAILS (FINE\_RANGE) ,

CONSTRAINT CHK\_DATE\_ISSUE CHECK ( DATE\_ISSUE < DATE\_RETURN )

)

Insert into LMS\_BOOK\_ISSUE (BOOK\_ISSUE\_NO,MEMBER\_ID,BOOK\_CODE,DATE\_ISSUE,DATE\_RETURN,DATE\_RETURNED,FINE\_RANGE) values (1,'LM001','BL000001',to\_date('01-05-12','DD-MM-RR'),to\_date('16-05-12','DD-MM-RR'),to\_date('16-05-12','DD-MM-RR'),null)

Insert into LMS\_BOOK\_ISSUE (BOOK\_ISSUE\_NO,MEMBER\_ID,BOOK\_CODE,DATE\_ISSUE,DATE\_RETURN,DATE\_RETURNED,FINE\_RANGE) values (2,'LM002','BL000002',to\_date('01-05-12','DD-MM-RR'),to\_date('06-05-12','DD-MM-RR'),to\_date('16-05-12','DD-MM-RR'),'R2')

Insert into LMS\_BOOK\_ISSUE (BOOK\_ISSUE\_NO,MEMBER\_ID,BOOK\_CODE,DATE\_ISSUE,DATE\_RETURN,DATE\_RETURNED,FINE\_RANGE) values (3,'LM003','BL000007',to\_date('01-04-12','DD-MM-RR'),to\_date('16-04-12','DD-MM-RR'),to\_date('20-04-12','DD-MM-RR'),'R1')

1. **Application of Database:-**

**Some cases**

1. To retrieve column on the basis of category java from table

SELECT \* FROM LMS\_BOOK\_DETAILS

WHERE CATEGORY=’JAVA’;

1. To sort the column values in ascending or descending order

SELECT MEMBER\_ID, MEMBER\_NAME AS Member\_name\_in\_Ascending\_order, DATE\_REGISTER, DATE\_EXPIRE

FROM LMS\_MEMBERS

ORDER BY MEMBER\_NAME DESC;

1. To group the data on the basis of membership status column

SELECT MEMBERSHIP\_STATUS, COUNT(MEMBER\_ID) FROM LMS\_MEMBERS

GROUP BY MEMBERSHIP\_STATUS;

1. To find out books which price greater than 350 rupees

SELECT CATEGORY, SUM(PRICE) FROM LMS\_BOOK\_DETAILS

WHERE PRICE>350

GROUP BY CATEGORY;

1. To find ‘JAVA’ category books that are over 500 rupees

SELECT BOOK\_TITLE, PRICE, CATEGORY FROM LMS\_BOOK\_DETAILS

WHERE CATEGORY=’JAVA’ AND PRICE>500;

1. To find out count of book code greater than 350 rupees

SELECT COUNT(BOOOK\_CODE), PRICE, CATEGORY FROM LMS\_BOOK\_DETAILS

WHERE PRICE>350

GROUP BY CATEGORY;

1. Using user defined function to count the total members

CREATE OR REPLACE FUNCTION totalmember

RETURN NUMBER IS

total INT;

BEGIN

SELECT COUNT(\*) INTO total

FROM LMS\_MEMBERS;

RETURN total;

END;

SELECT totalmember() // call function

1. Making list of book code category basis

SELECT CATEGORY,

LISTAGG(BOOK\_CODE) AS BOOK\_LIST FROM LMS\_BOOK\_DETAILS;

1. To find out the count of book code and sum of price category basis

SELECT CATEGORY , COUNT(BOOK\_CODE) AS count\_book, SUM(PRICE) AS sum\_price FROM LMS\_BOOK\_DETAILS,

GROUP BY CATEGORY;

1. To find out top or first value and price in descending order

SELECT \* FROM

(

SELECT CATEGORY, BOOK\_TITLE, PRICE, FIRST\_VALUE(BOOK\_TITLE)OVER(PARTITION BY CATEGORY

ORDER BY PRICE DESC) AS top\_price

FROM LMS\_BOOK\_DETAILS )

WHERE BOOK\_TITLE = top\_price;

1. Find average price on rows between 2 preceding and current rows

SELECT CATEGORY, BOOK\_TITLE, PRICE , AVG(PRICE) OVER(PARTITION BY CATEGORY ORDER BY BOOK\_TITLE

ROWS BETWEEEN 2 PRECEDING AND CURRENT ROW)

FROM LMS\_BOOK\_DETAILS;

1. To find out member id which fine amount lesser than 50

SELECT MEMBER\_ID FROM LMS\_BOOK\_ISSUE A INNER JOIN LMS\_FINE\_DETAILS B

ON A.FINE\_RANGE = B.FINE\_RANGE GROUP BY MEMBER\_ID

HAVING SUM (FINE\_AMOUNT) <50;

1. Create CTE to find average price in descending order book category wise

WITH avg\_category\_price AS

(

SELECT CATEGORY, AVG(PRICE) AS Avg\_price FROM .MS\_BOOK\_DETAILS

GROUP BY CATEGORY

)

SELECT \* FROM avg\_category\_price ORDER BY avg\_price DESC;

1. To check the book in category if present then 1 else 0 in all books

SELECT CATEGORY,

CASE WHEN CATEGORY=’JAVA’ THEN 1 ELSE 0 END AS CATEGORY\_JAVA,

CASE WHEN CATEGORY=’ALGORITHMS’ THEN 1 ELSE 0 END AS CATEGORY\_ALGORITHMS,

1. To count the total number of a book in category

SELECT

SUM(CASE WHEN CATEGORY=’JAVA’ THEN 1 ELSE 0 END) AS CATEGORY\_JAVA,

SUM(CASE WHEN CATEGORY=’C’ THEN 1 ELSE 0 END) AS CATEGORY\_C,

SUM(CASE WHEN CATEGORY=’FUNDAMENTAL’ THEN 1 ELSE 0 END) AS CATEGORY\_FUNDAMENTAL,

SUM(CASE WHEN CATEGORY=’Python’ THEN 1 ELSE 0 END) AS CATEGORY\_Python,

SUM(CASE WHEN CATEGORY=’C#’ THEN 1 ELSE 0 END) AS CATEGORY\_C#,

SUM(CASE WHEN CATEGORY=’ALGORITHMS’ THEN 1 ELSE 0 END) AS CATEGORY\_ALGORITHMS,

SUM(CASE WHEN CATEGORY=’DATA.STRUCTURES’ THEN 1 ELSE 0 END) AS CATEGORY\_DATA.STRUCTURES,

FROM LMS\_BOOK\_DETAILS;

1. Find running total for fine paid

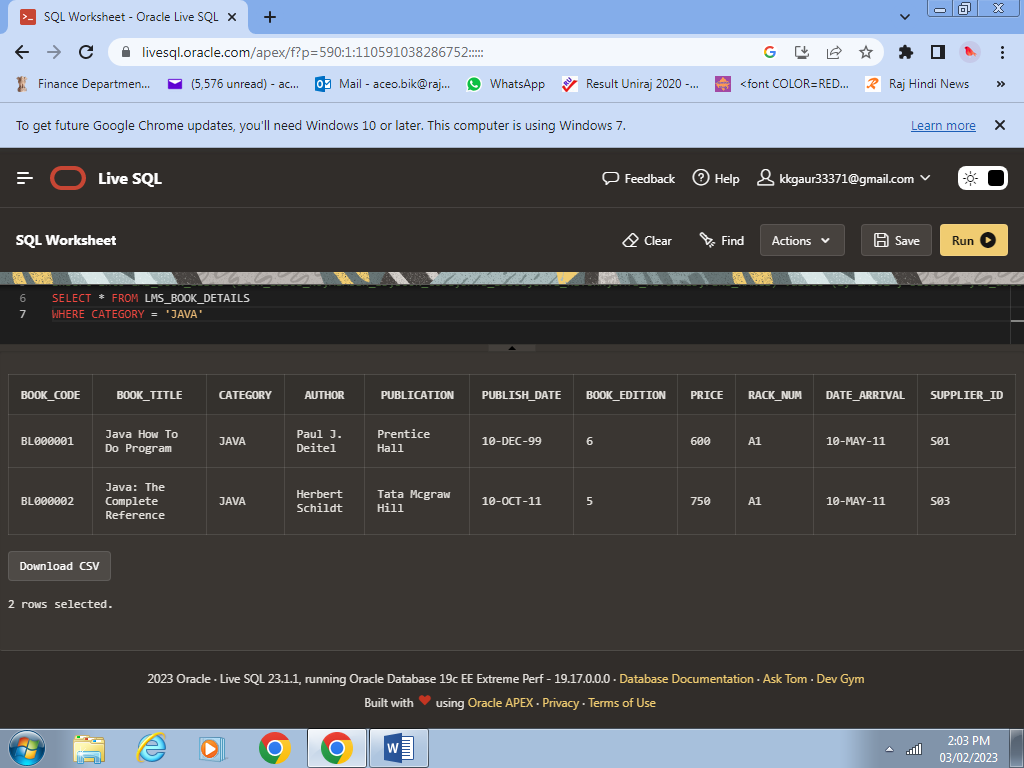
SELECT MEMBER\_ID, RUNNING\_TOTAL\_FINE FROM

(

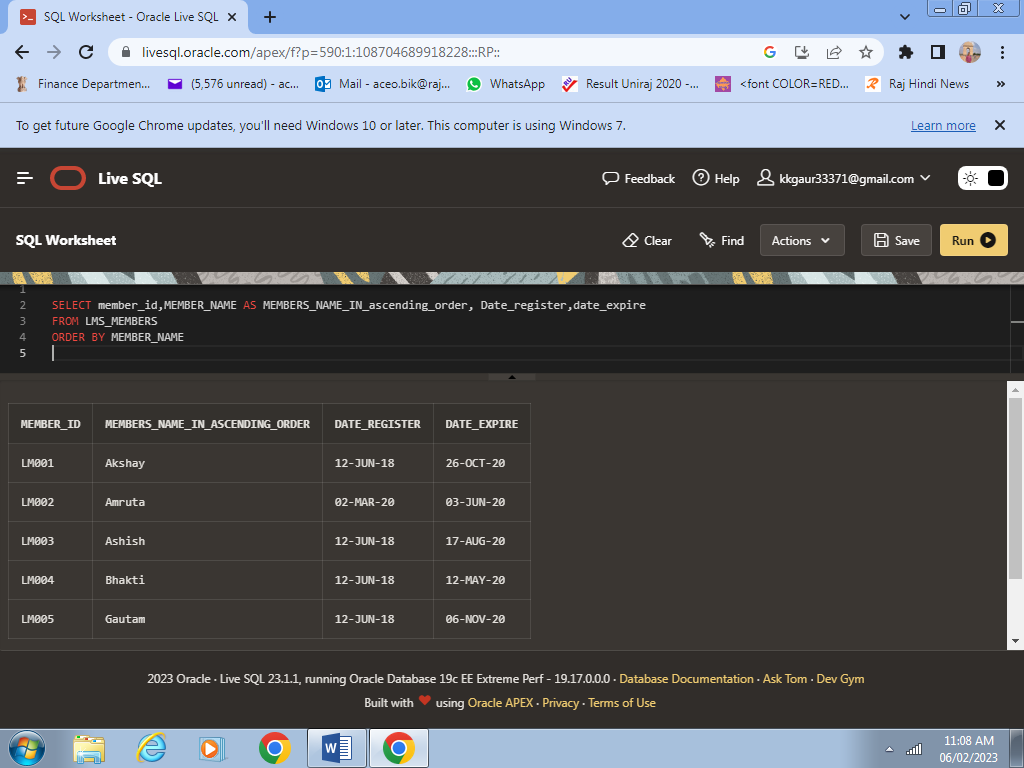
SELECT MEMBER\_ID, FINE\_AMOUNT, SUM(FINE\_AMOUNT) OVER (ORDER BY MEMBER\_ID ASC )AS RUNNING\_TOTAL\_FINE FROM LMS\_BOOK\_ISSUE I INNER JOIN LMS\_FINE\_DETAILS D

ON I. FINE\_RANGE = D. FINE\_RANGE )

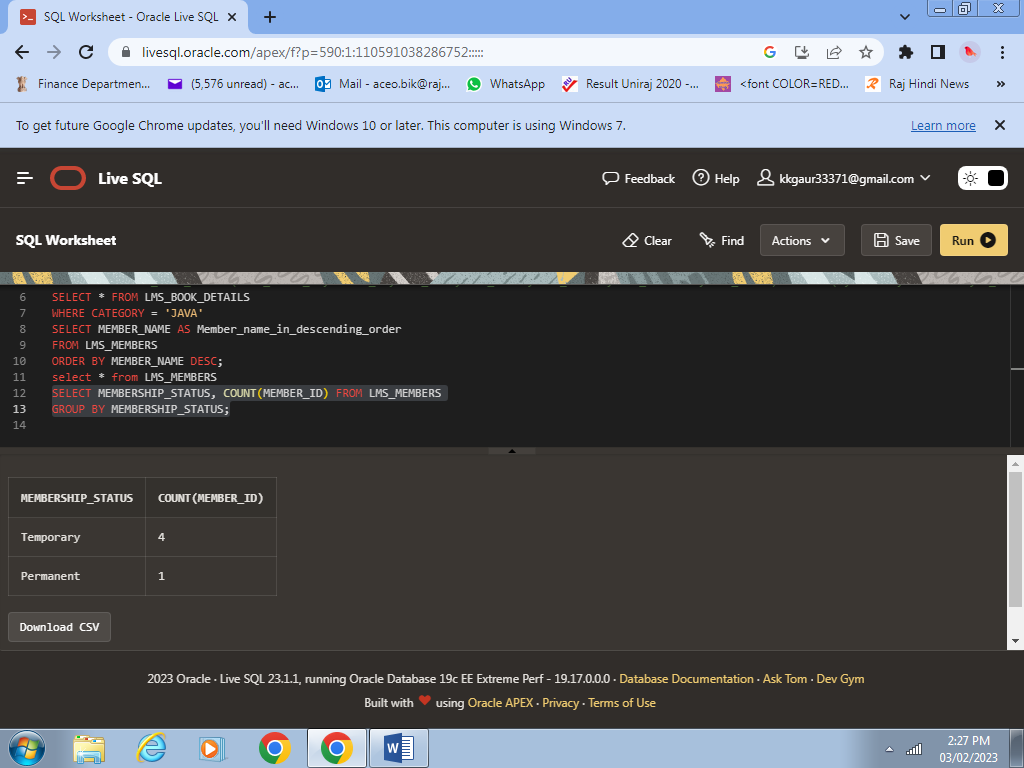
1. **Representation:-**



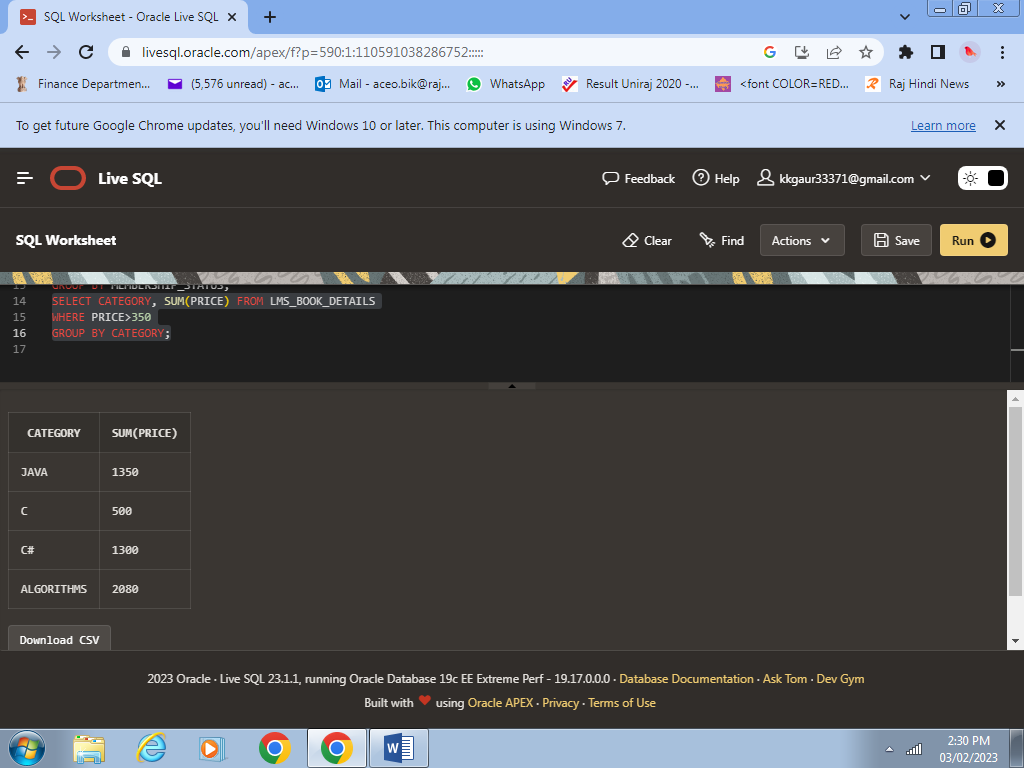
Case 1



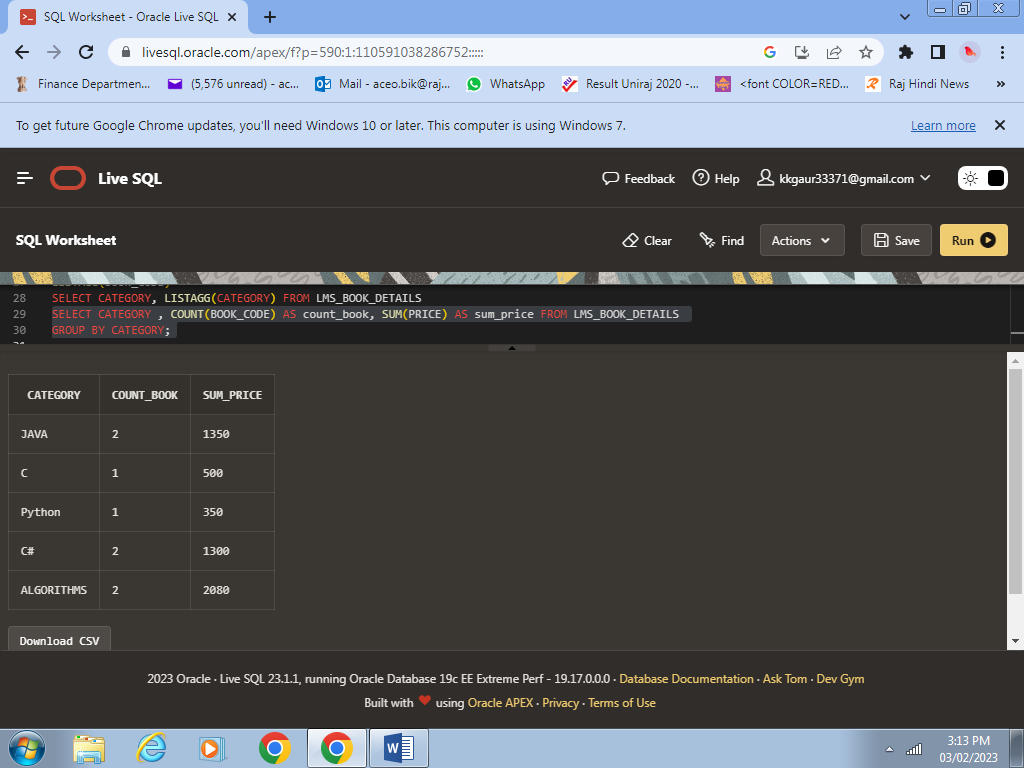
Case 2



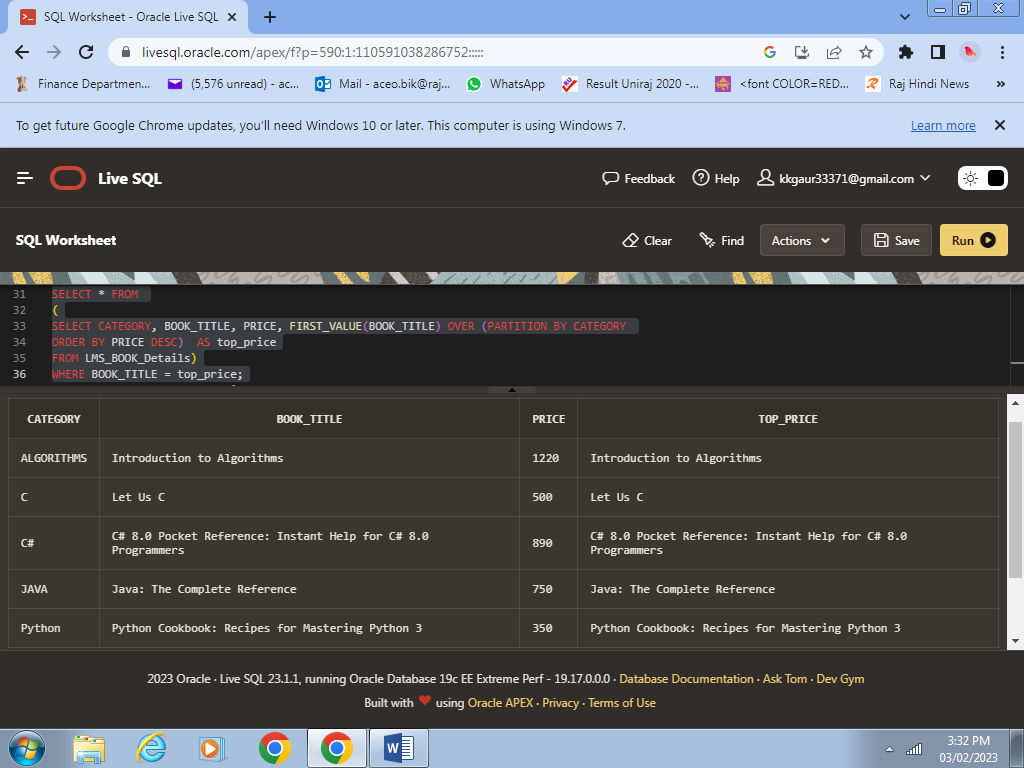
Case 3



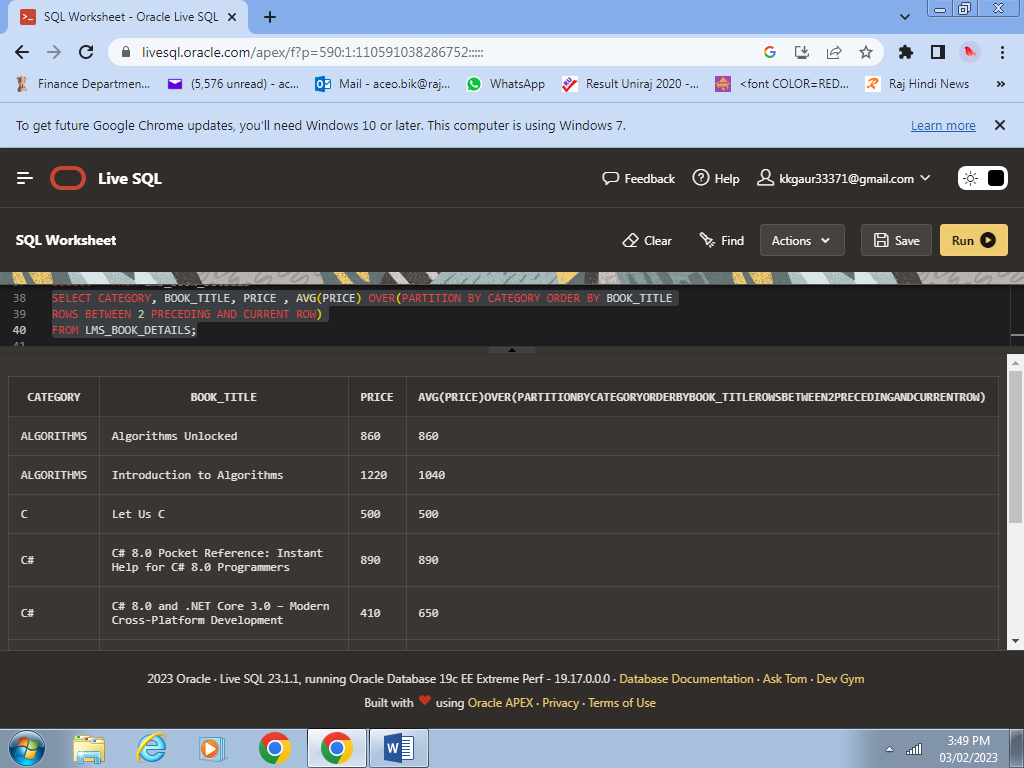
Case 4



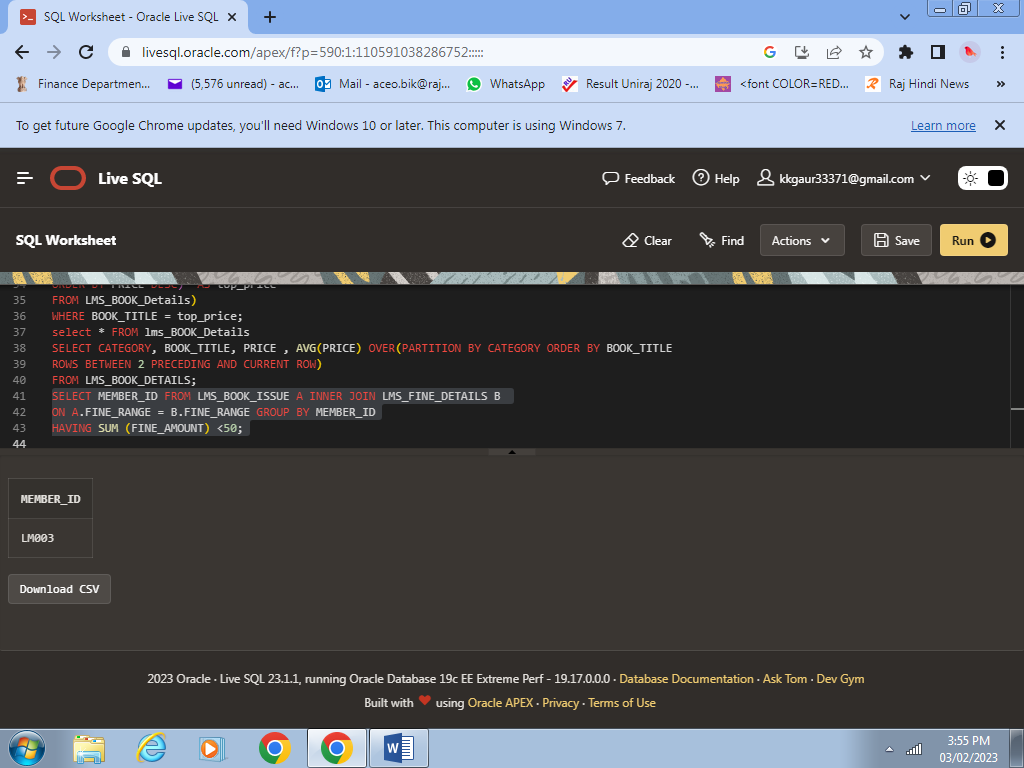
CASE 9



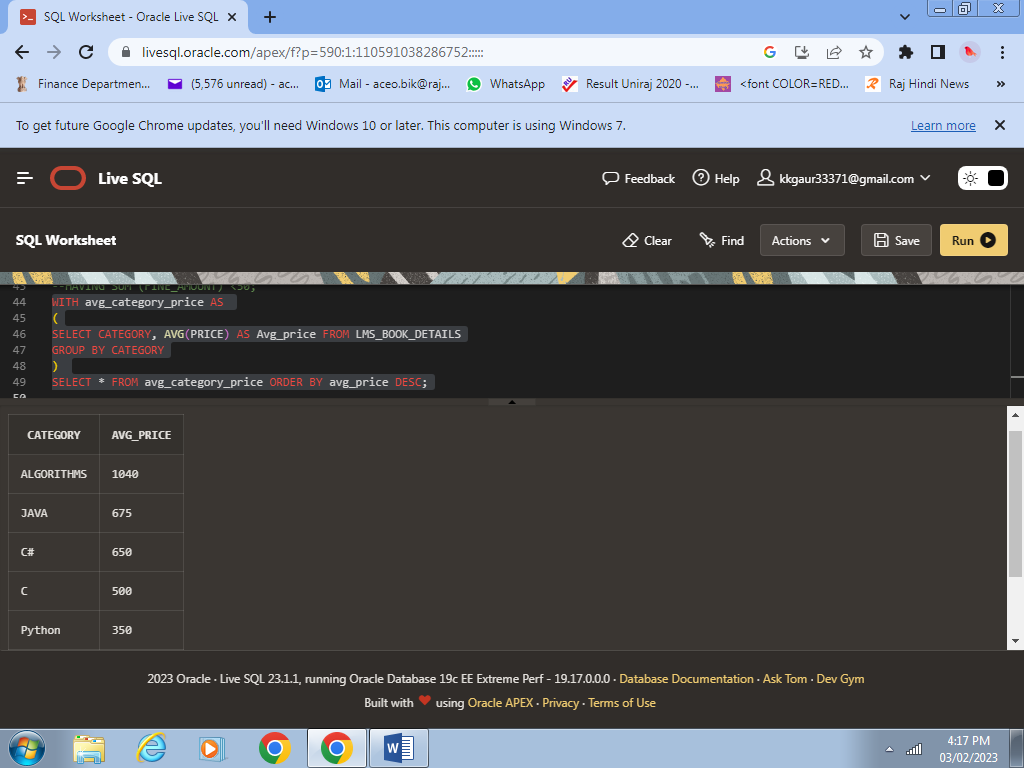
Case 10



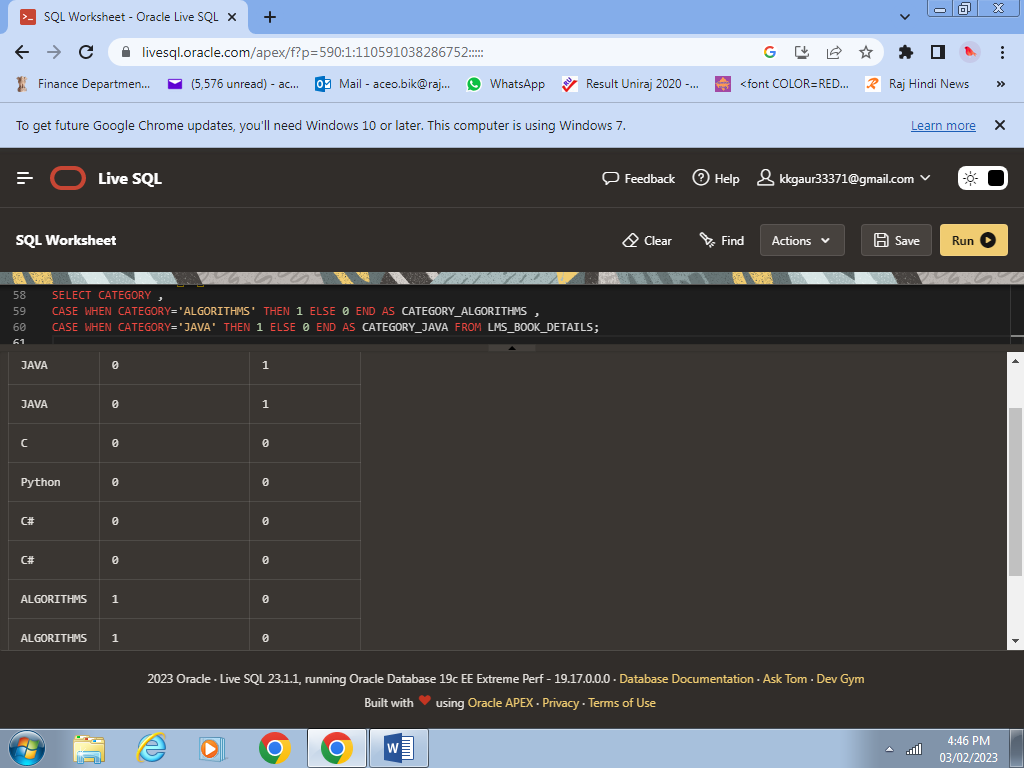
Case 11



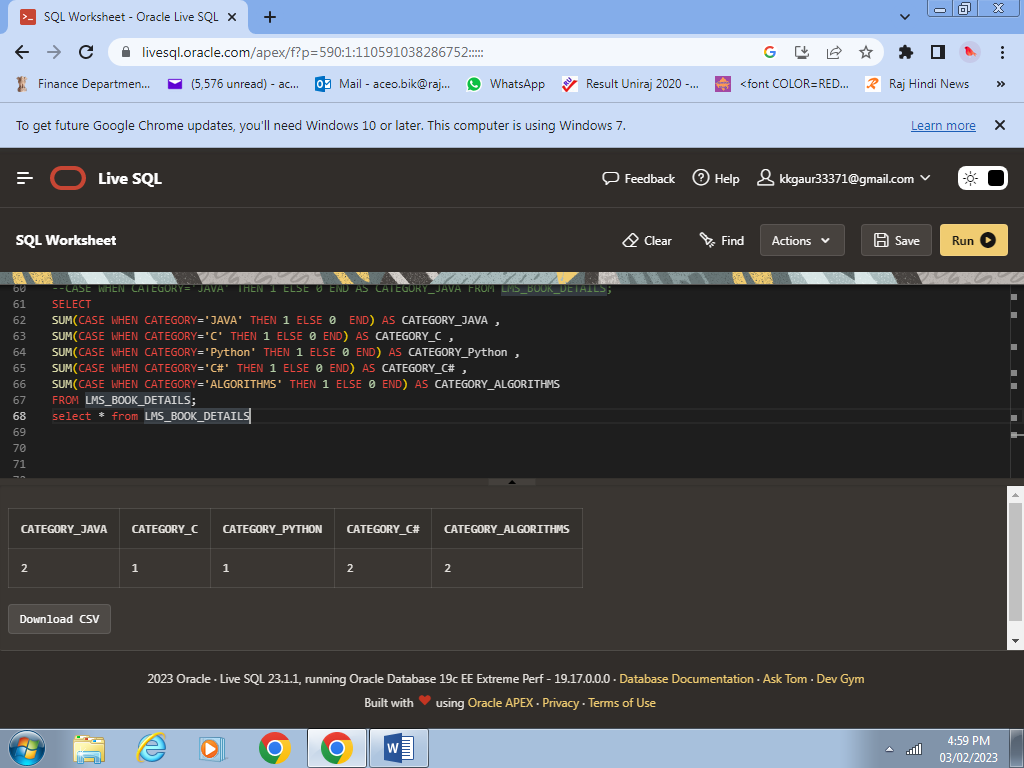
Case 12



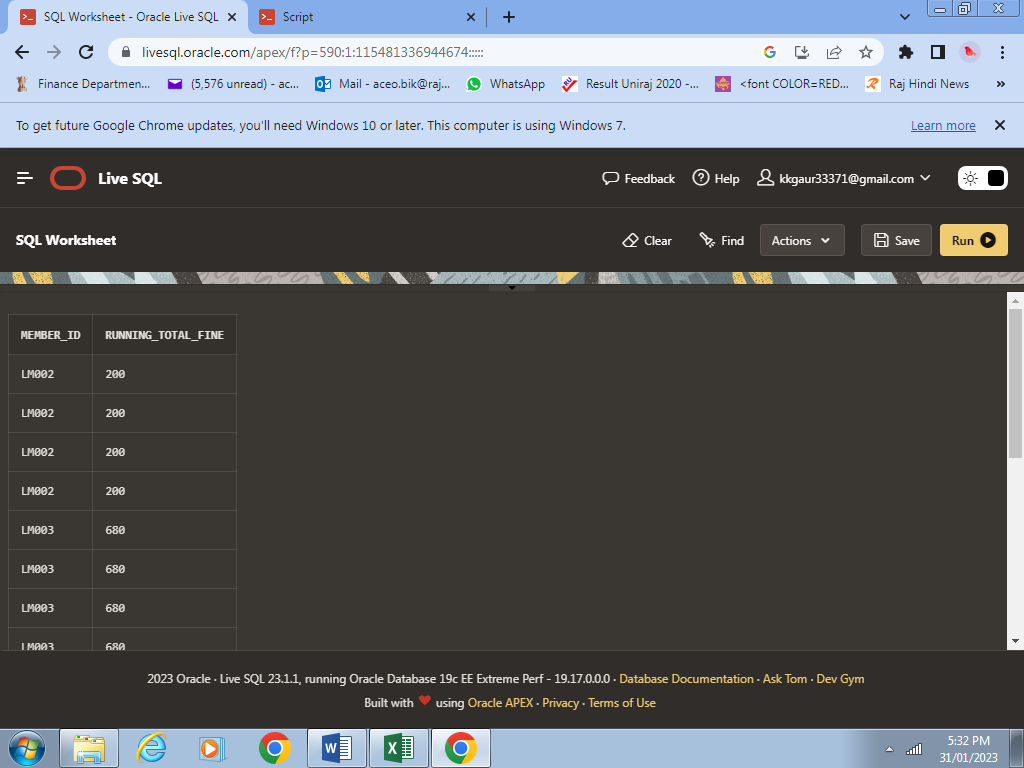
Case 13



Case 14



Case 15



Case 16

1. **Result:-**

Managed to successfully create the LIBRARY MANAGEMENT DATABASE

USING ORACLE SQL and it was user friendly and cost effective.

Helpful in library management.

1. **Conclusion:-**

It can be concluded that LIBRARY MANAGEMENT DATABASE was a success.

This database created using SQL. It is user friendly and cost effective.

Low complexity and provide better services.