**DELHI TECHNOLOGICAL UNIVERSITY**

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Bawana Road, Delhi – 110042

**DEPARTMENT OF SOFTWARE ENGINEERING**



**Advanced Database Management System Lab File**

Subject Code: SE-318

**Submitted to Submitted by**

Mr. Ram Murti Rawat Ankur Yadav

Dept. of Software Engineering 2K20/SE/18

INDEX

|  |  |  |  |
| --- | --- | --- | --- |
| **SNO.** | **TOPIC** | **DATE** | **SIGN** |
| **1** | Overview of SQL | 12/01/2023 |  |
| **2** | Study ER Diagram | 19/01/2023 |  |
| **3** | Implement DML Commands | 02/02/2023 |  |
| **4** | Study and Implement Integrity Constraints | 09/02/2023 |  |
| **5** | Implement Aggregate Functions | 23/02/2023 |  |
| **6** | Implement string and numeric functions | 09/03/2023 |  |
| **7** | Write queries to implement joins | 23/03/2023 |  |
| **8** | Implement SQL Correlated queries | 06/04/2023 |  |
| **9** | Write SQL Queries to implement WITH Clause | 13/04/2023 |  |
| **10** | Write a program to study PL/SQL | 20/04/2023 |  |

**EXPERIMENT – 1**

**AIM:** Overview of SQL

**DDL (Data Definition Language) Commands**

DDL or Data definition language is actually the definition or description of the database structure or schema, it won't change the data inside the database. Create, modify, and delete the database structures, but not the data. Only These commands are not done by all the users, who have access to the database via an application.

**CREATE Command in SQL**

SQL Create the database or its object (ie table, index, view, function, etc.).

Syntax

CREATE DATABASE databasename



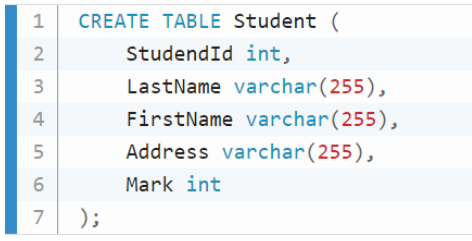
Syntax

CREATE TABLE table\_name (

column1 datatype,

column2 datatype,

column3 datatype, .... );

****

**DROP Command in SQL**

Drop command helps to delete the object from the database (ie table, index, view, function, etc.).

Syntax

DROP object object\_name

****

Syntax

DROP DATABASE database\_name

****

**ALTER Command in SQL**

Alter command is helpful to change or modify the structure of the database or its object.

Syntax

ALTER TABLE table\_name

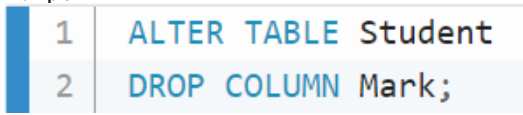
ADD column\_name datatype

****

Syntax

ALTER TABLE table\_name

DROP COLUMN column\_name

****

**DCL (Data Control Language) Commands**

DCL or Data Control Language is to provide rights, permissions, and other controls of the database system.

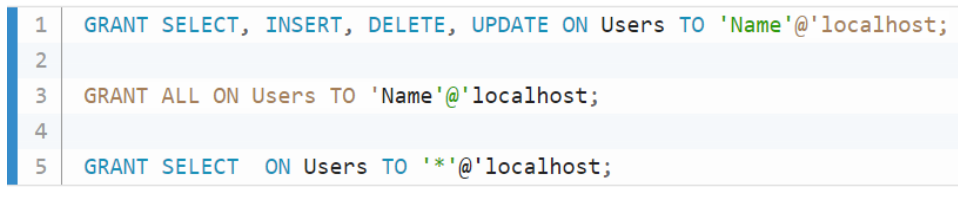
**GRANT Command in SQL**

GRANT command is helpful to provide privileges to the database.

Syntax

GRANT privileges\_names

ON object TO user

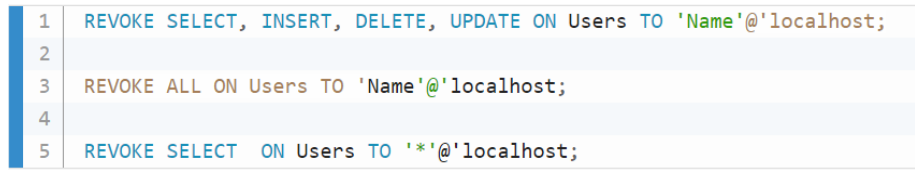
****

**REVOKE Command in SQL**

SQL Revoke command is to withdraw the user’s access privileges given by using the GRANT command.

Syntax

REVOKE privileges ON object FROM user

****

**DML (Data Manipulation Language) Commands**

DML or Data Manipulation Language is to manipulate the data inside the database. With the help of DML commands, we can insert, delete, and change the data inside the database

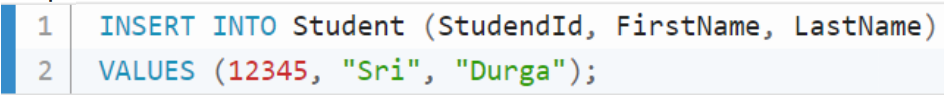
**INSERT Command in SQL**

SQL Insert command is helpful to insert the data into a table. 1) All the column names are mentioned in the insert statement.

Syntax

INSERT INTO table\_name (column1, column2, column3, ...)

VALUES (value1, value2, value3, ...)

****

Syntax

INSERT INTO table\_name

VALUES (value1, value2, value3, ...)

****

**UPDATE Command in SQL**

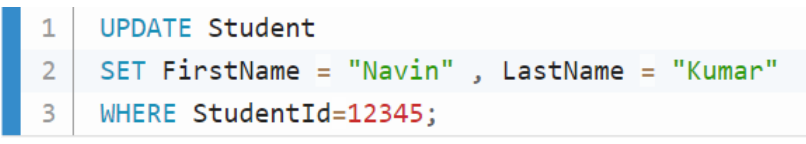
SQL Update command is helpful to update the existing data in a table.

Syntax

UPDATE table\_name

SET column1 = value1, column2 = value2, ...

WHERE condition

****

**Delete Command in SQL**

SQL Delete command helps to delete the records from a database table.

Syntax

DELETE FROM table\_name

WHERE condition;

****

**LOCK Command in SQL**

SQL Lock command is helpful to lock the table to control concurrency.

Syntax

LOCK TABLE table-Name

IN { SHARE | EXCLUSIVE } MODE

****

**EXPERIMENT – 2**

**Aim:**

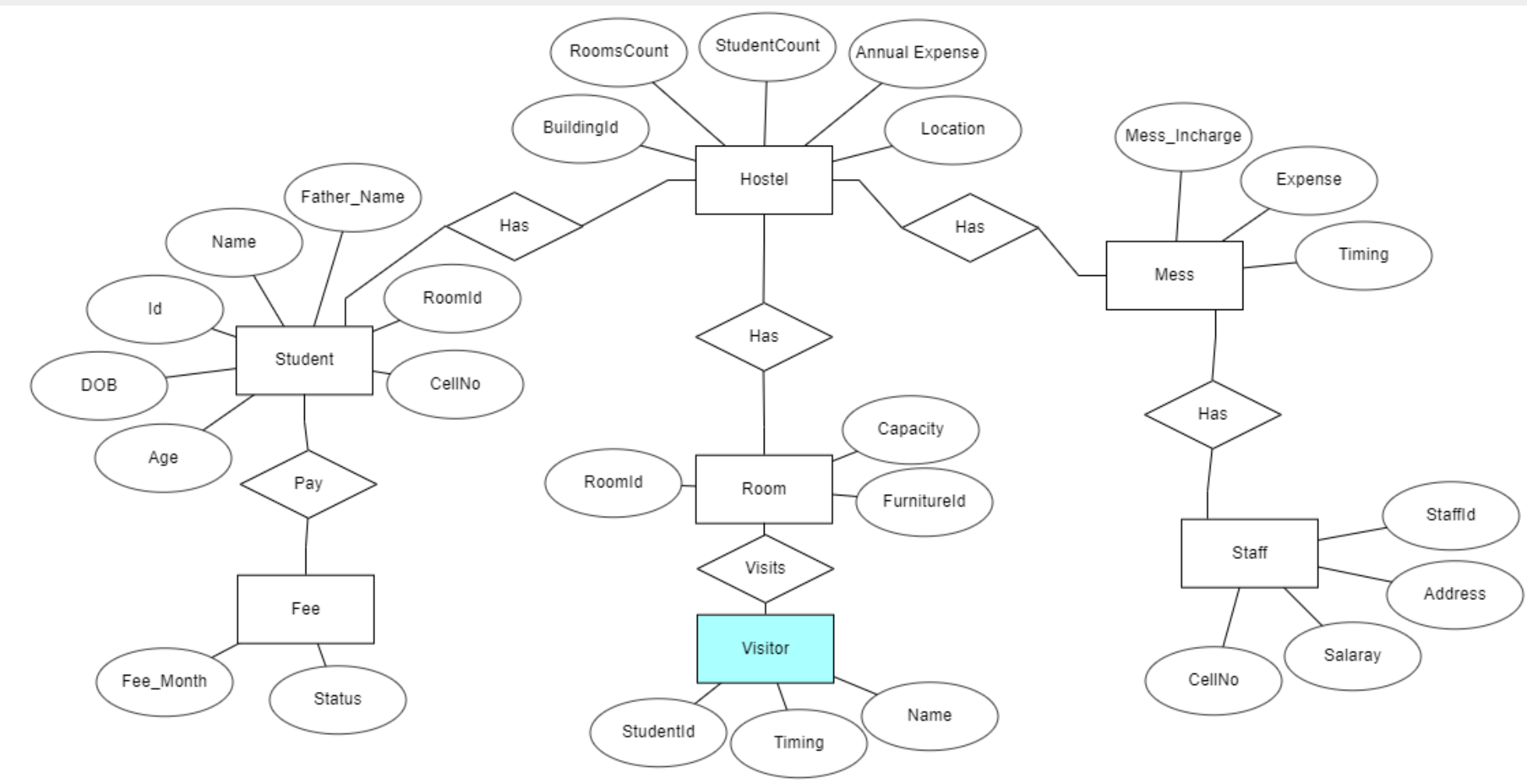
To draw the Entity Relationship Diagram (ERD) for the Restaurant management system.

**Theory:**

Entity Relationship Diagram is a type of structural diagram for use in database design. An ERD contains different symbols and connectors that visualize two important pieces of information:

* The major entities within the system scope
* The inter-relationships among these entities

**ER Diagram:**



**Conclusions:**

Learned about the construction of entity relationship diagram and its basic knowledge.

**EXPERIMENT – 3**

**Aim:**

To run Data Manipulation Language commands (DML) for Hospital Management System.

**Theory:**

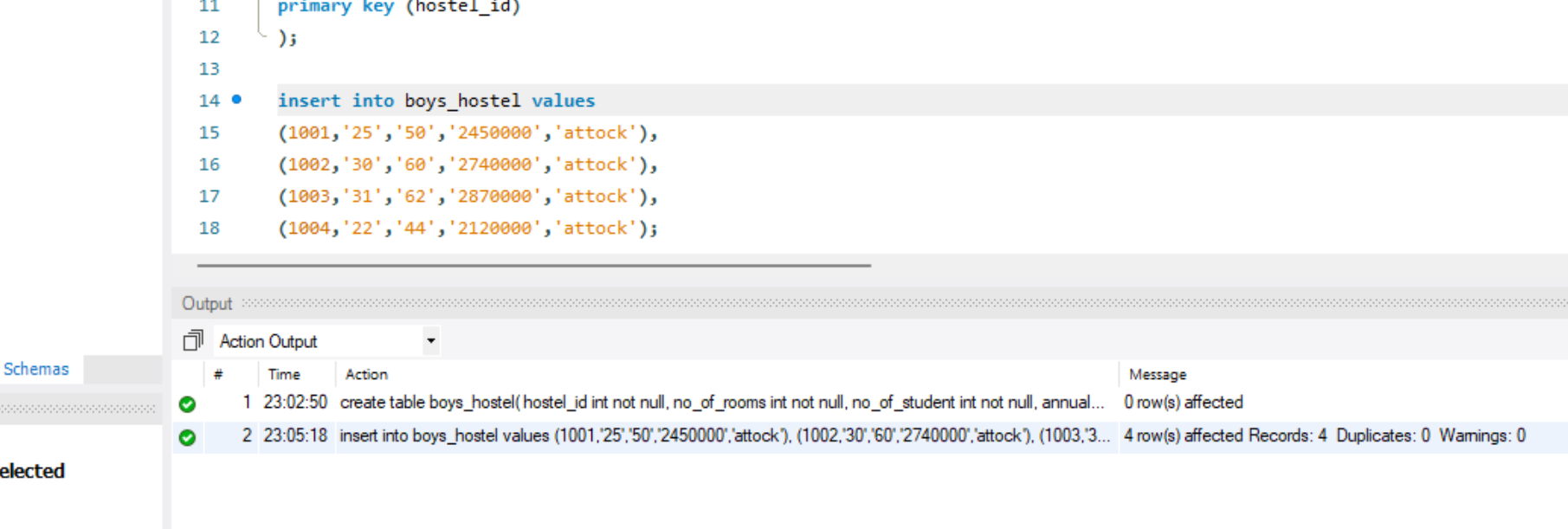
The structured query language (SQL) commands deal with the manipulation of data present in the database that belongs to the DML or Data Manipulation Language. This includes most of the SQL statements. It is the component of the SQL statement that controls access to data and to the database.

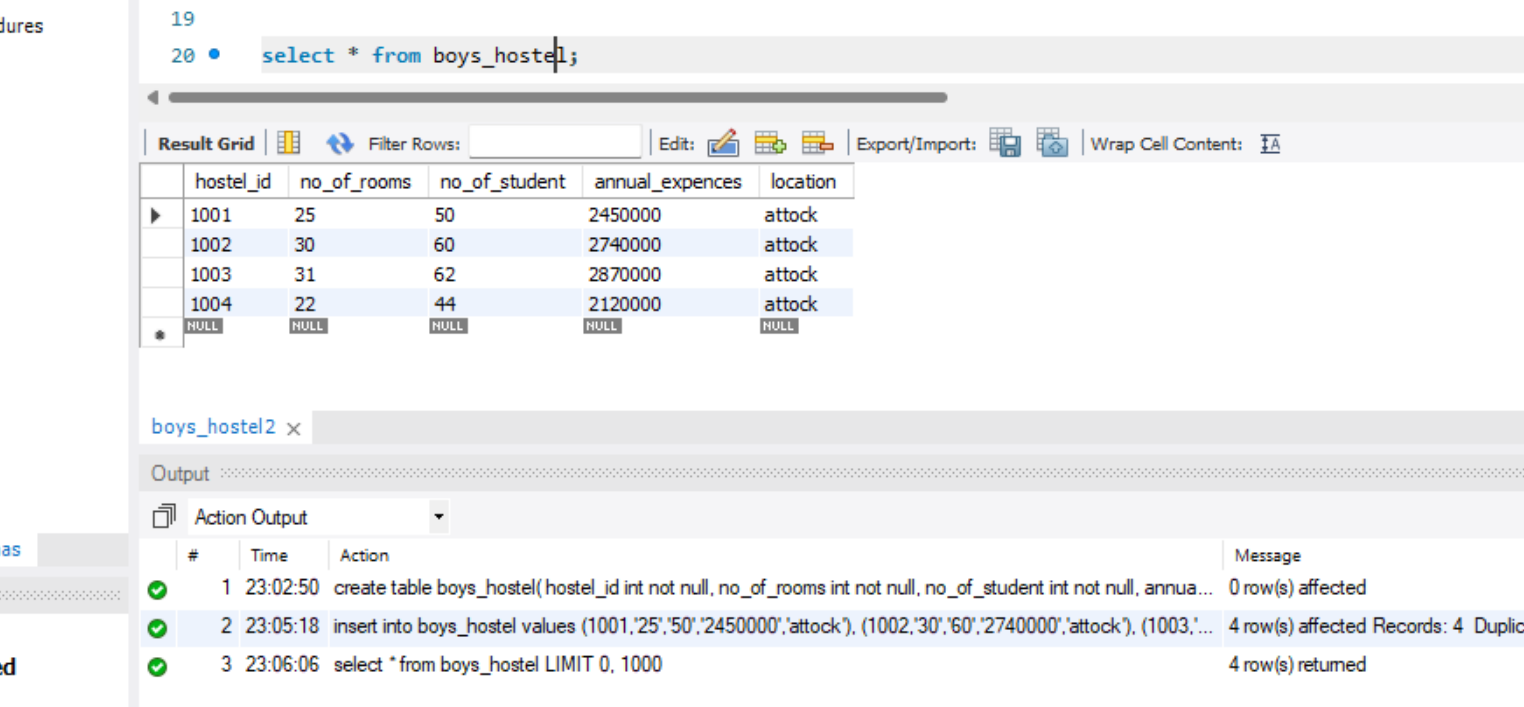
List of DML commands:

* **INSERT**: It is used to insert data into a table.
* **UPDATE**: It is used to update existing data within a table.
* **DELETE**: It is used to delete records from a database table.

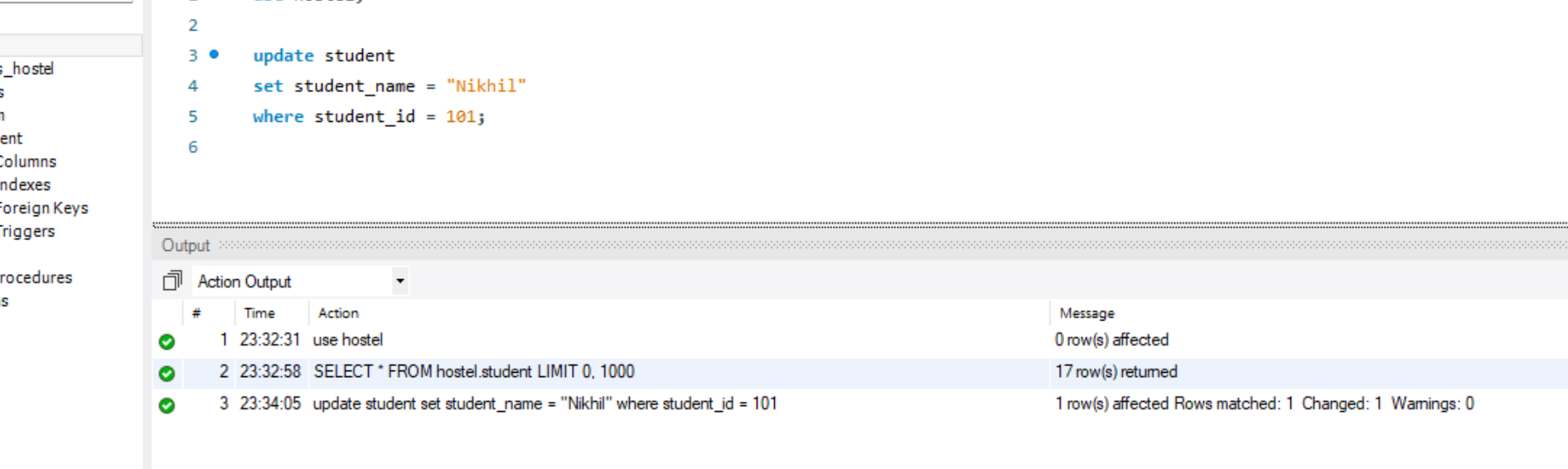
**Queries:**

SQL QUERIES FOR INSERT AND SELECT COMMANDS –

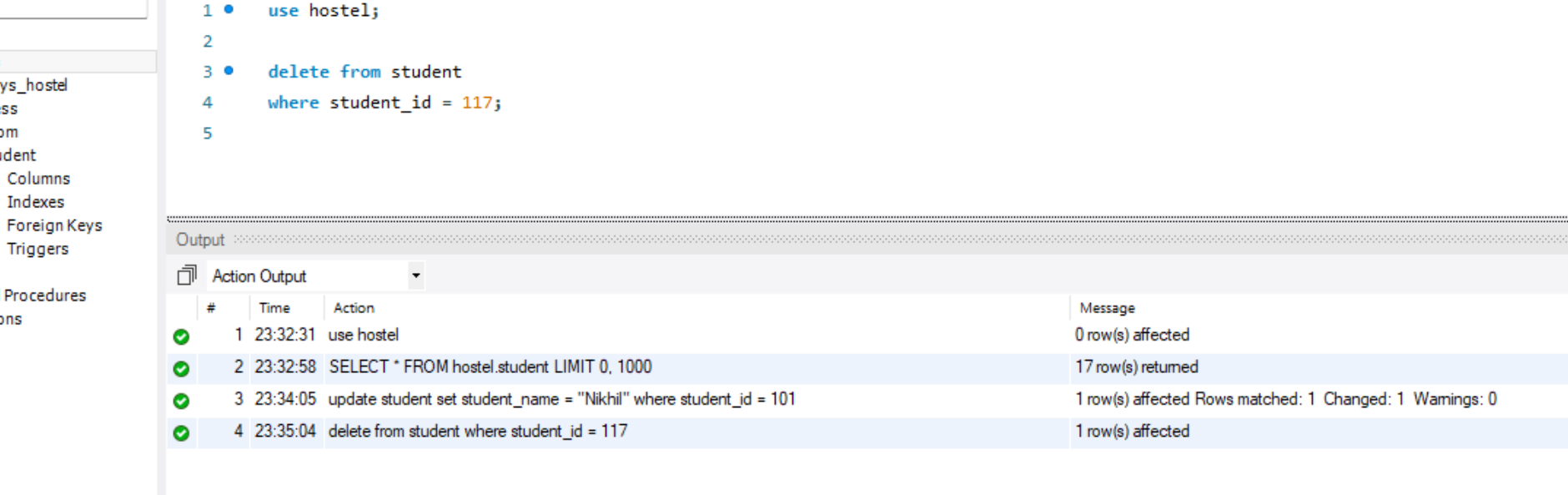




SQL QUERIES FOR UPDATE COMMANDS –



SQL QUERIES FOR DELETE COMMANDS –



**Conclusion:**

Learned about the basics and syntax of data manipulation language commands (dml).

**EXPERIMENT – 4**

**Aim:**

To study and implement integrity constraints for employee management system.

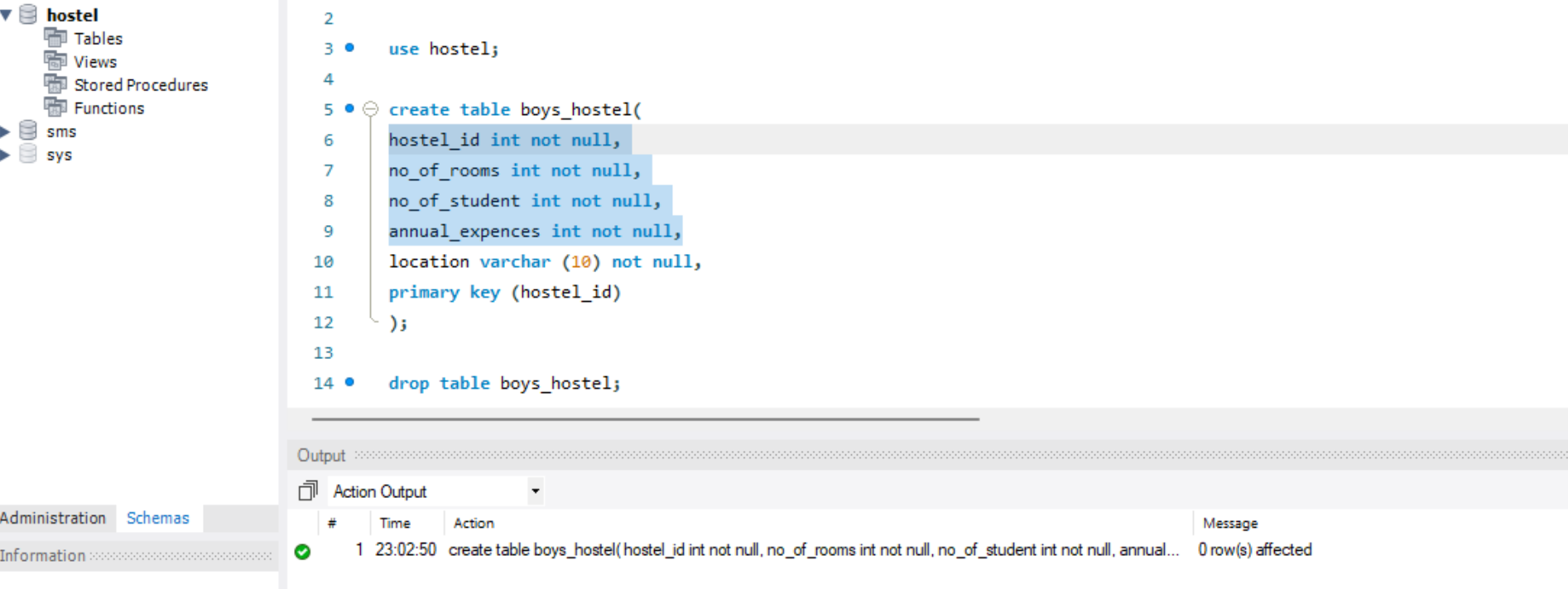
**Theory:**

Integrity Constraints are applied at the column or table and used to restrict the types of information that can be entered into a table. This means that the data in the database is accurate and reliable.

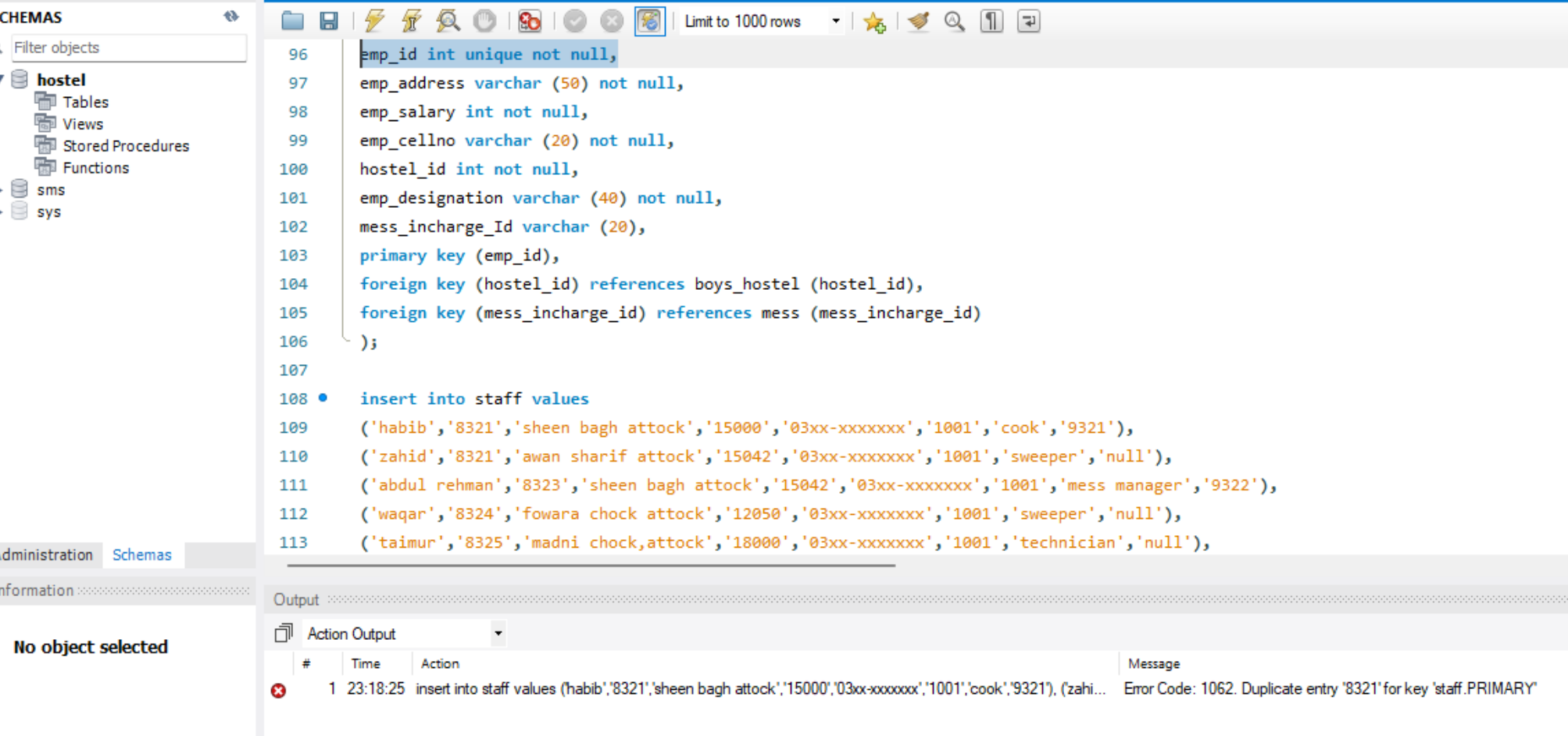
1. **NOT NULL** - ensures that a column cannot have a NULL value.
2. **UNIQUE** - ensures that all values in a column are different.
3. **PRIMARY KEY** - combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table.
4. **FOREIGN KEY** - prevents actions that would destroy links between tables.

**Queries:**

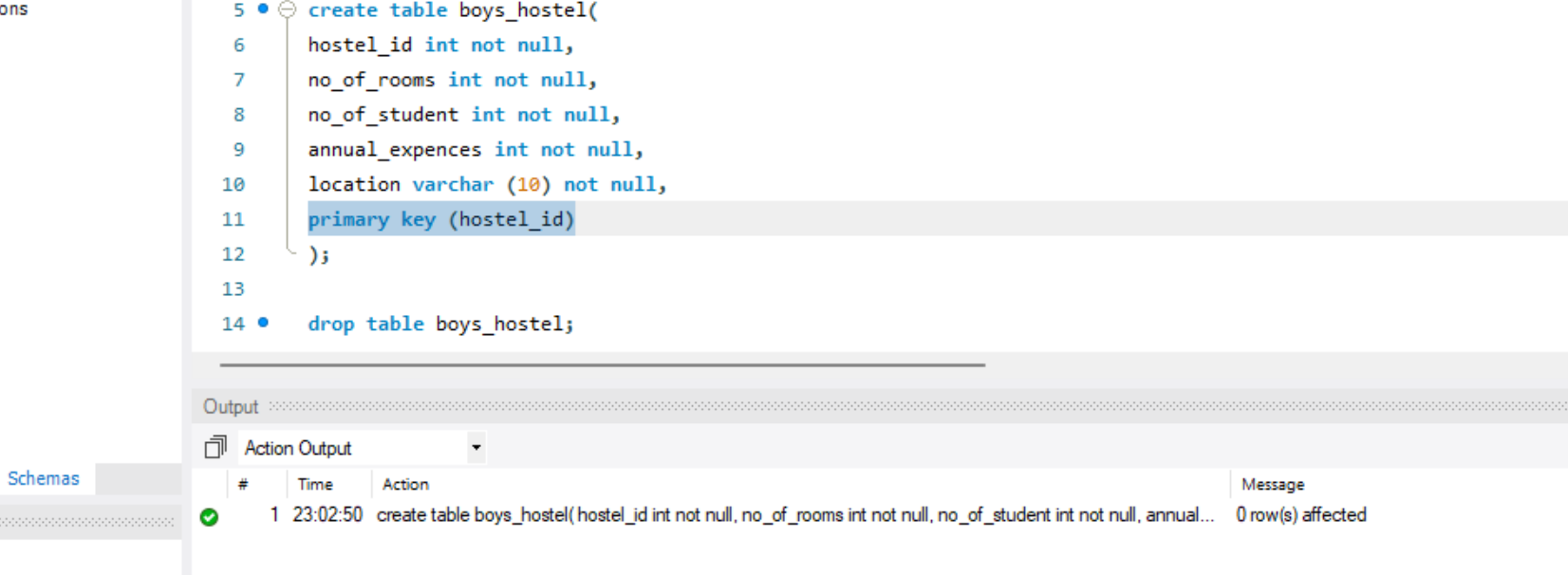
FOR NOT NULL –



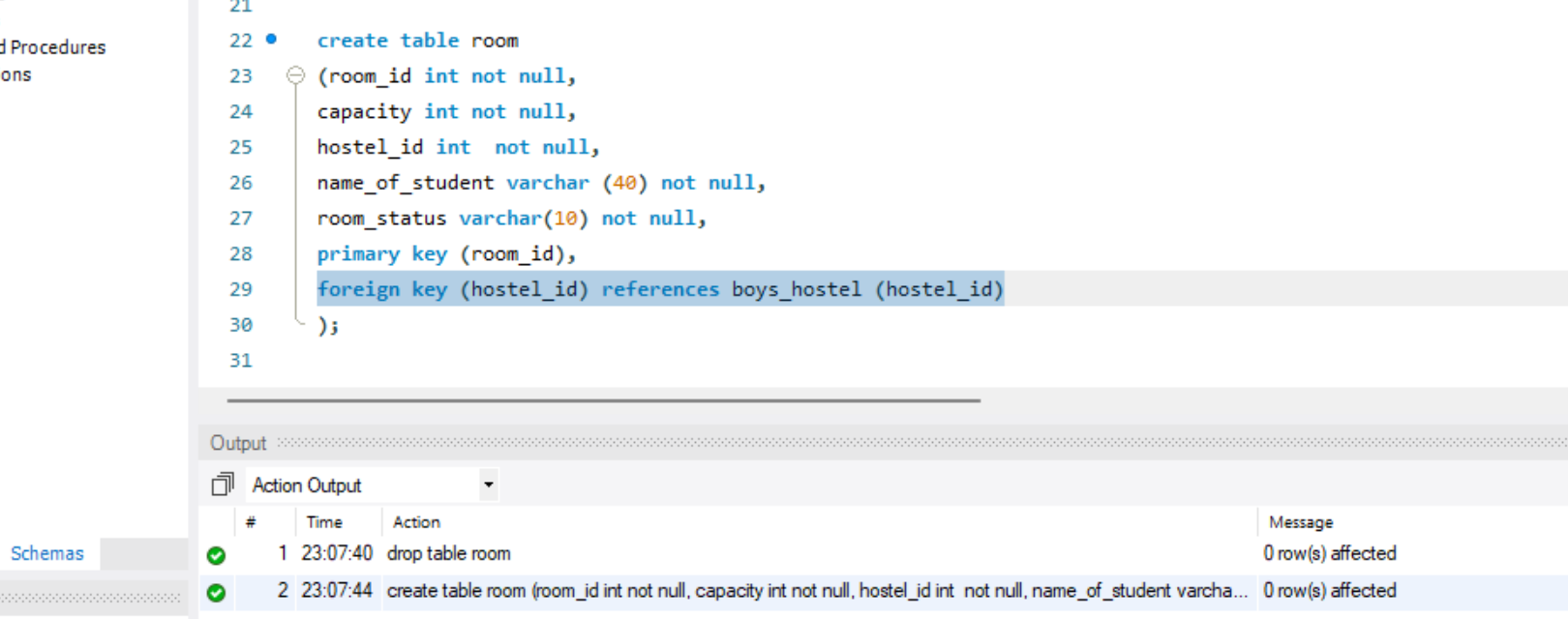
FOR UNIQUE –



FOR PRIMARY KEY –



FOR FOREIGN KEY –



**Conclusions:**

Successfully, studied and implemented the integrity constraints for the Hospital management system.

**EXPERIMENT – 5**

**Aim:**

To study and implement aggregate functions for attendance management system.

**Theory:**

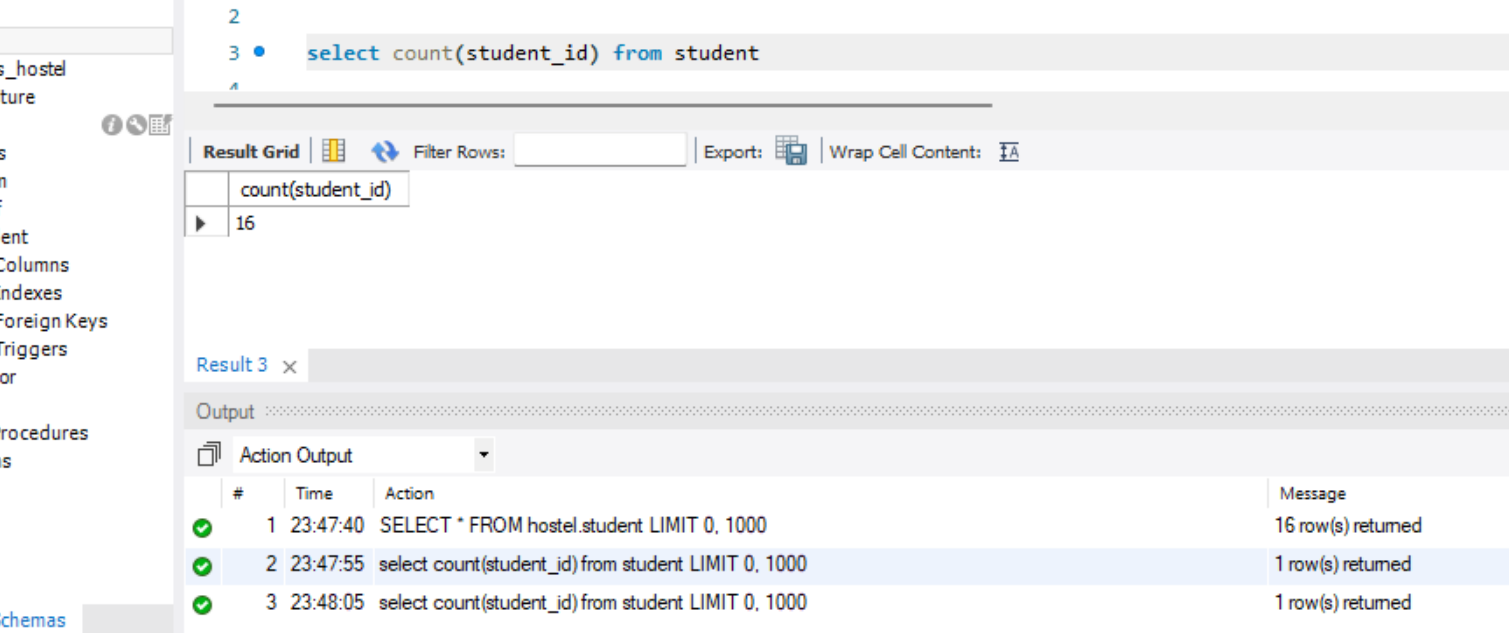
Aggregate functions in SQL are used to perform calculations on a set of values and return a single value as the result. These functions are used to summarize or group data in a table. There are several aggregate functions available in SQL, including COUNT, SUM, AVG, MAX, and MIN.

1. The COUNT function is used to count the number of rows in a table that meet a specific condition. It can be used with the \* symbol to count all the rows in the table, or with a specific column to count the number of rows where that column meets a specific condition.
2. The SUM function is used to calculate the sum of all the values in a column. It is used to find the total value of a column.
3. The AVG function is used to calculate the average value of a column. It is used to find the average value of a column.
4. The MAX and MIN functions are used to find the maximum and minimum values in a column, respectively. They are used to find the highest and lowest values in a column.

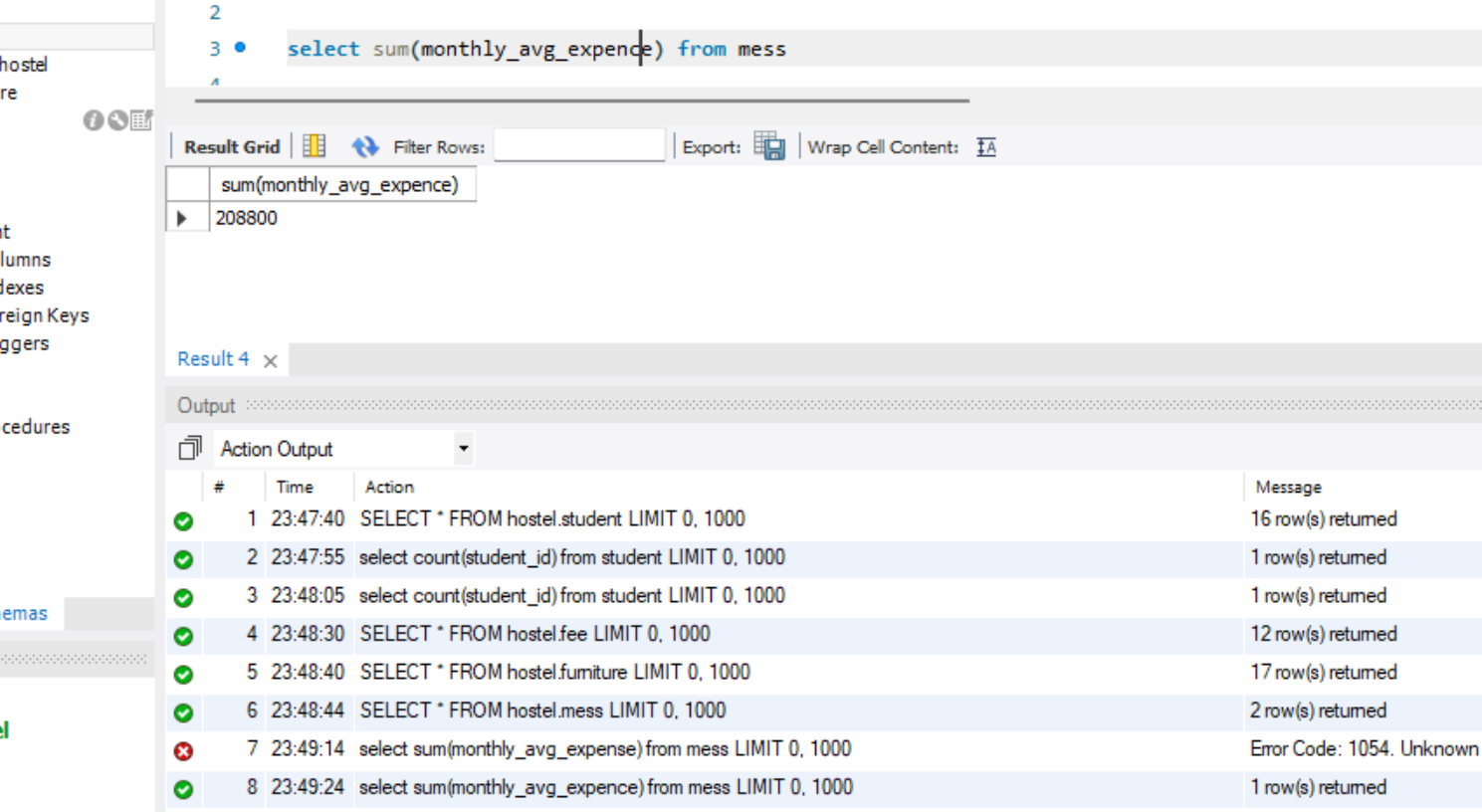
Aggregate functions can also be used with the GROUP BY clause to group the data by a specific column. This can be used to find the total, average, maximum, or minimum value of a column for each group in the table.

**Queries:**

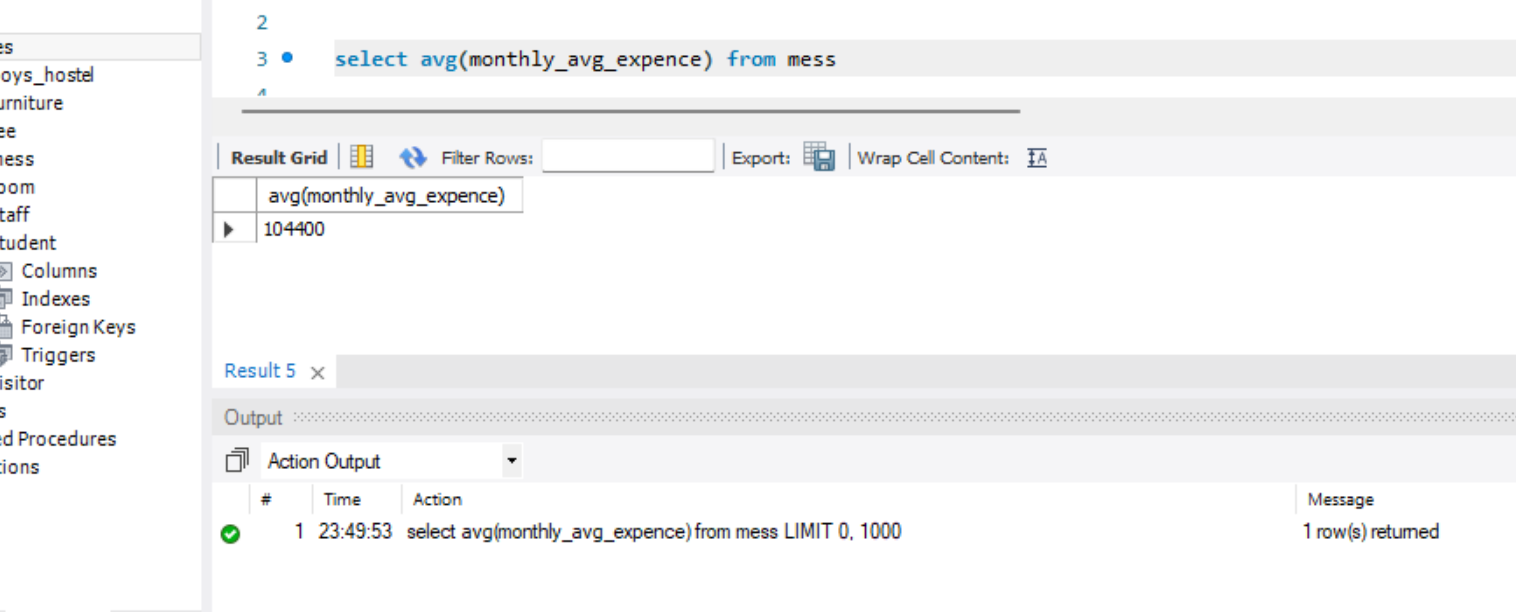
COUNT –



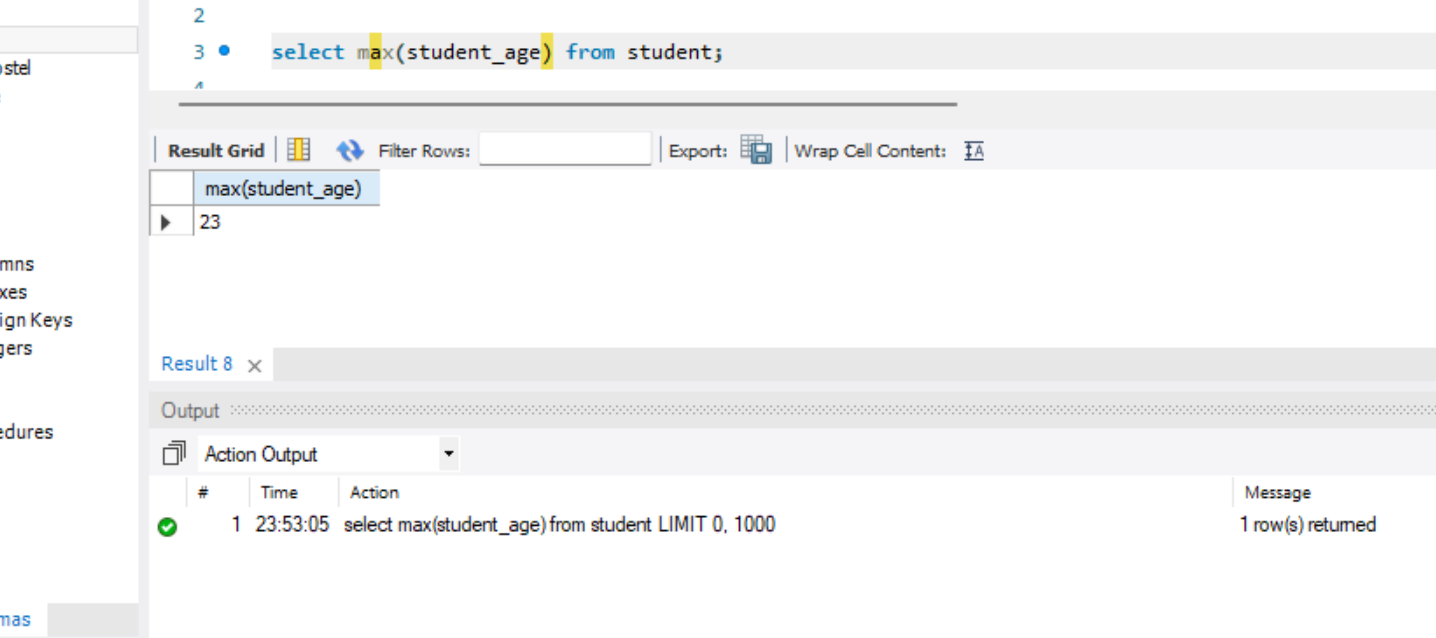
SUM –



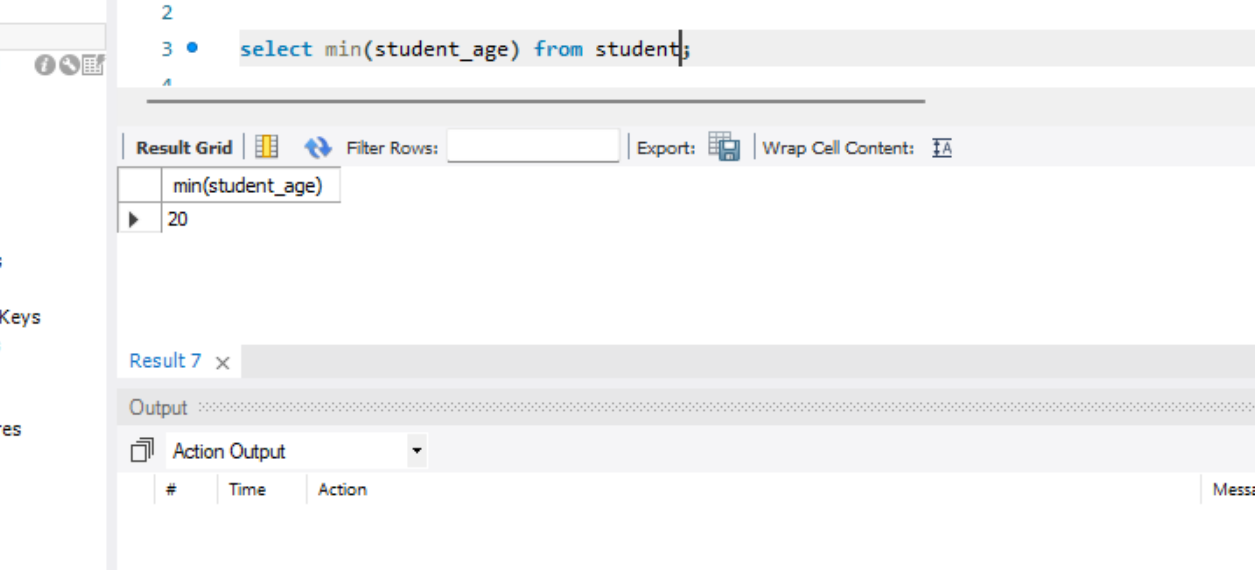
AVERAGE –



MAX –



MIN –



**Conclusions:**

Studied and implemented aggregate functions in SQL which are powerful tools that allow us to perform calculations on groups of data rather than individual rows. They are used to summarize and analyse large datasets, making it easier to extract useful information from them.

**EXPERIMENT – 6**

**Aim:**

To study and implement string and numeric functions.

**Theory:**

SQL String Functions

SQL String functions are the predefined functions that allow the database users for string manipulation. These functions only accept, process, and give results of the string data type.

1. ASCII
2. CONCAT
3. FORMAT
4. INSERT
5. REPLACE
6. REVERSE
7. SUBSTR
8. TRIM:
9. UCASE

SQL Numeric Functions

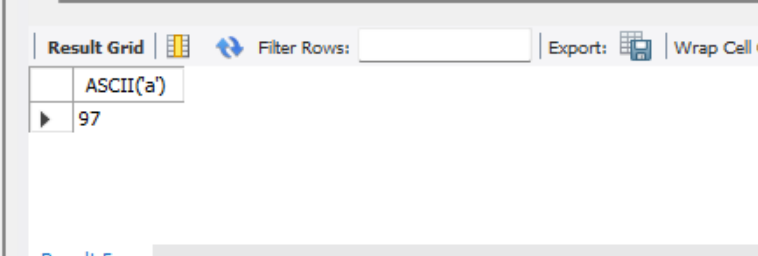
Numeric Functions are used to perform operations on number and return numbers. Following are the numeric functions defined in SQL:

1. ACOS
2. ATAN
3. GREATEST
4. LOG10
5. FLOOR
6. COUNT
7. DEGREES
8. TRUNCATE

**Queries:**

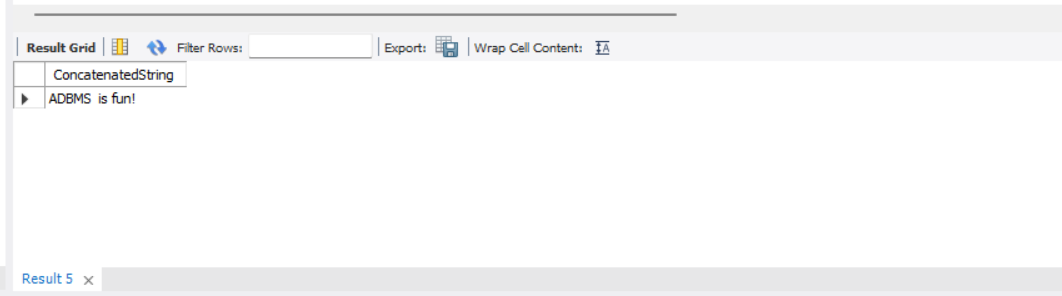
ASCII

SELECT ASCII('a');



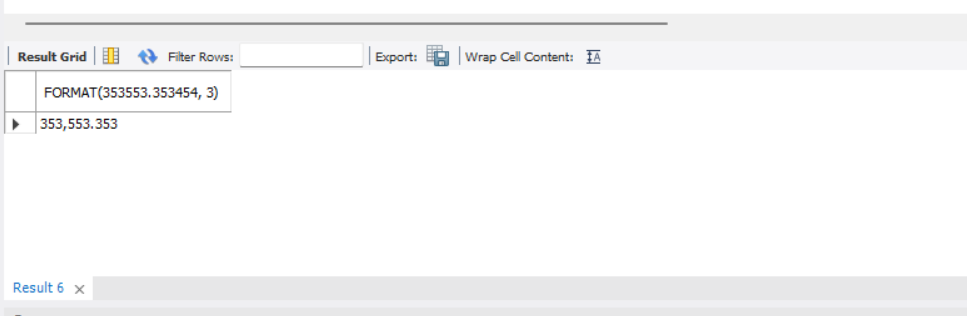
CONCAT

SELECT CONCAT("ADBMS ", " ", "is ", "fun!") AS ConcatenatedString;



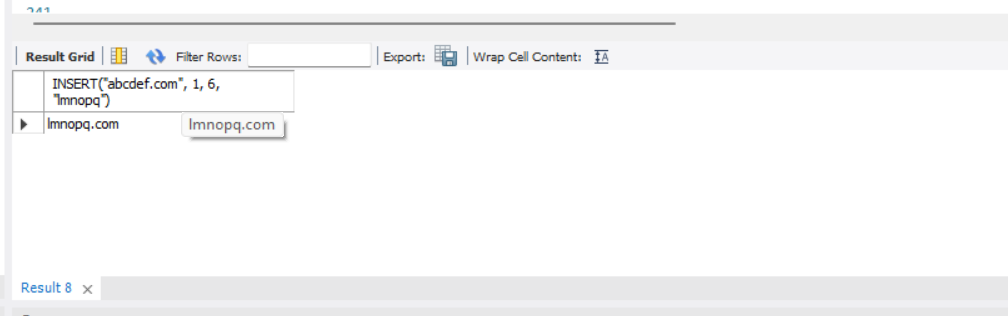
FORMAT

SELECT FORMAT(353553.353454, 3);



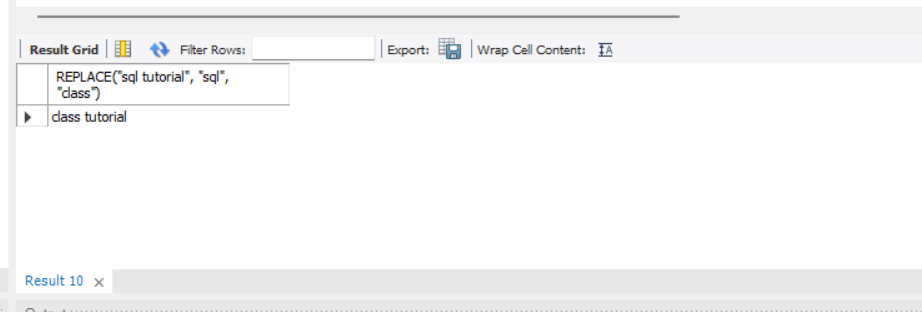
INSERT

SELECT INSERT("abcdef.com", 1, 6, "lmnopq");



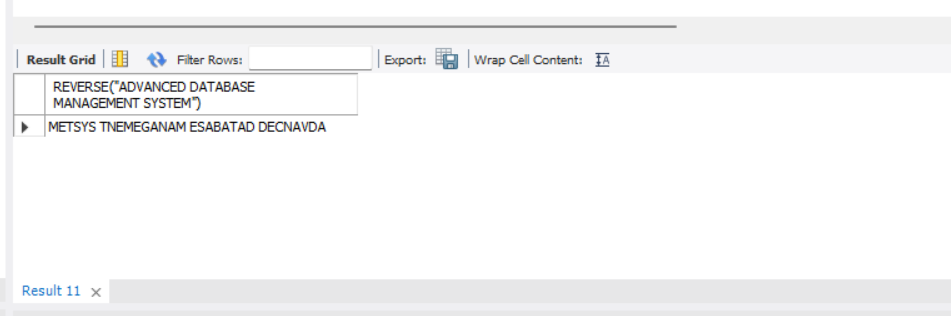
REPLACE

SELECT REPLACE("sql tutorial", "sql", "class");



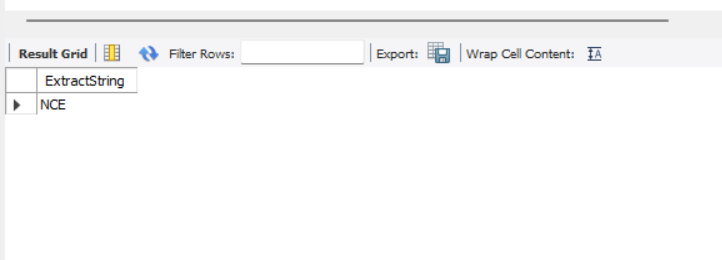
REVERSE

SELECT REVERSE("ADVANCED DATABASE MANAGEMENT SYSTEM");



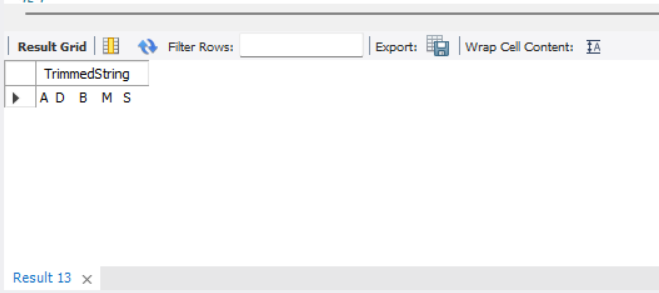
SUBSTER

SELECT SUBSTR("ADVANCED DATABASE", 5, 3) AS ExtractString;



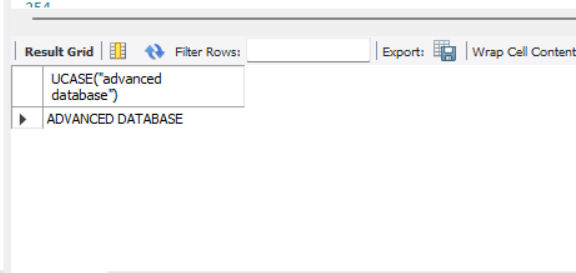
TRIM

SELECT TRIM(" A D B M S ") AS TrimmedString;



UCASE

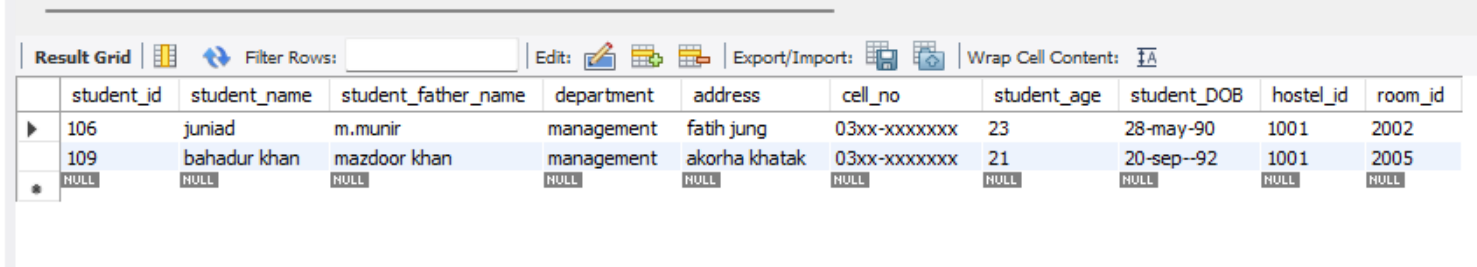
SELECT UCASE("advanced database");



LIKE

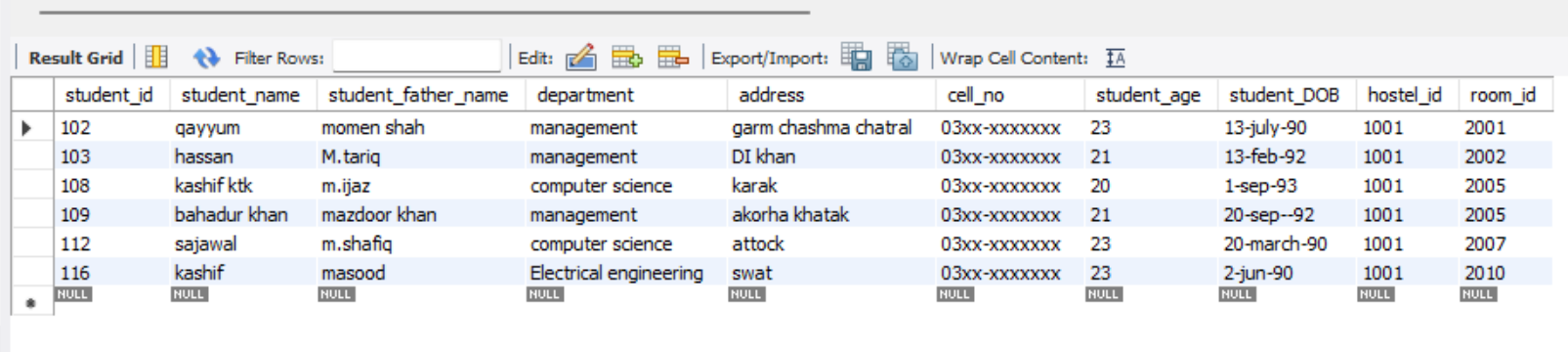
select \* from student

where student\_name like '%ad%'



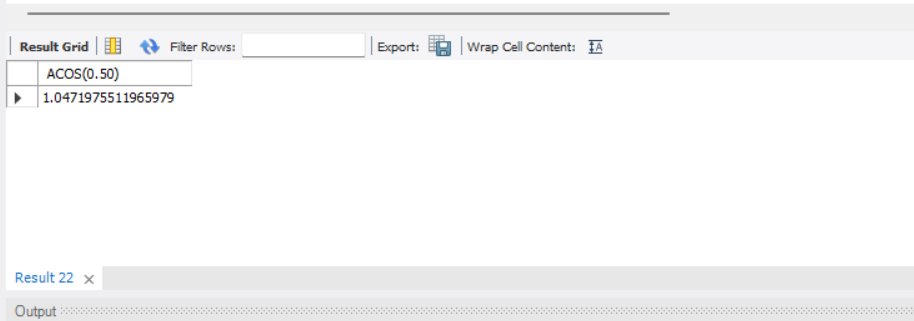
select \* from student

where student\_name like '\_a%'



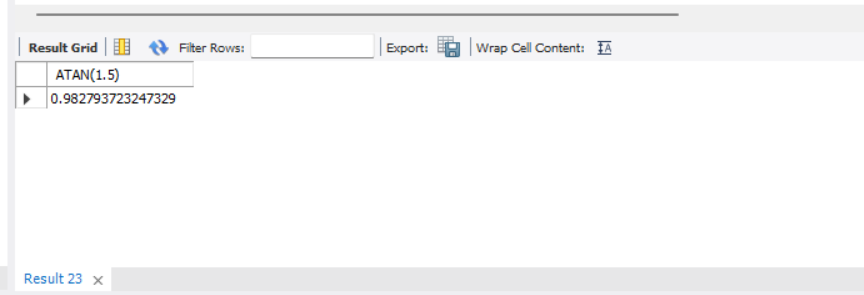
ACOS

SELECT ACOS(0.50);



ATAN

SELECT ATAN(1.5);



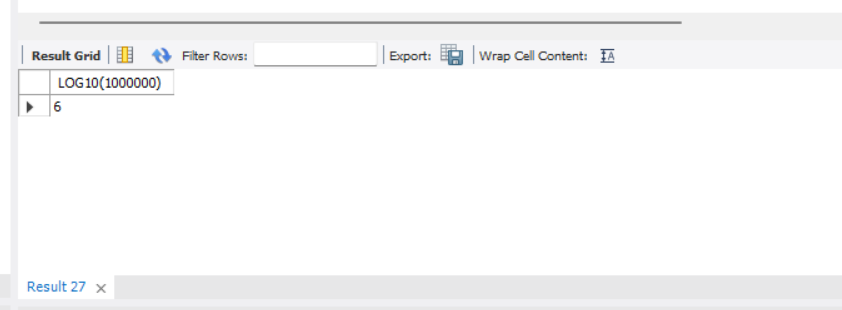
GREATEST

SELECT GREATEST(23, 898, 132, 13144, 133241, 13231);



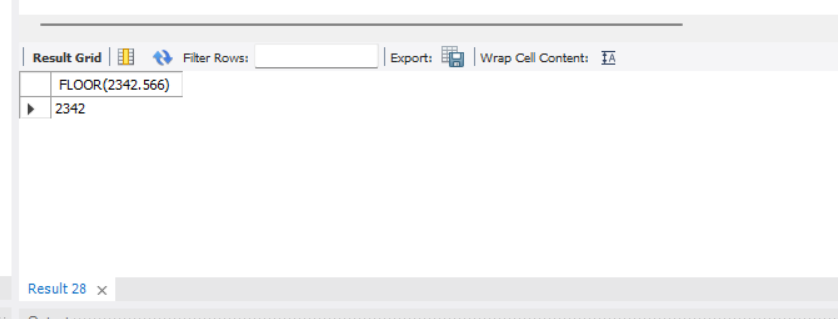
LOG10

SELECT LOG10(1000000);



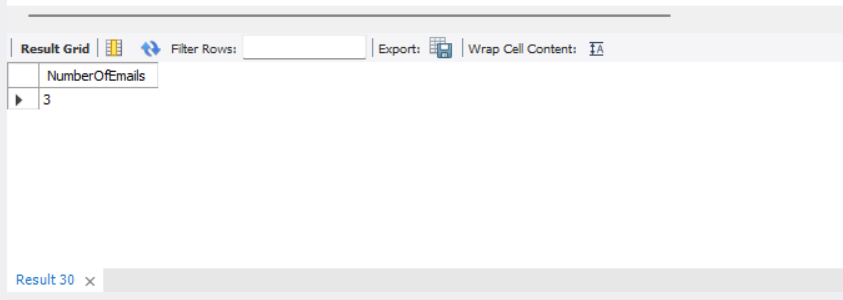
FLOOR

SELECT FLOOR(2342.566);



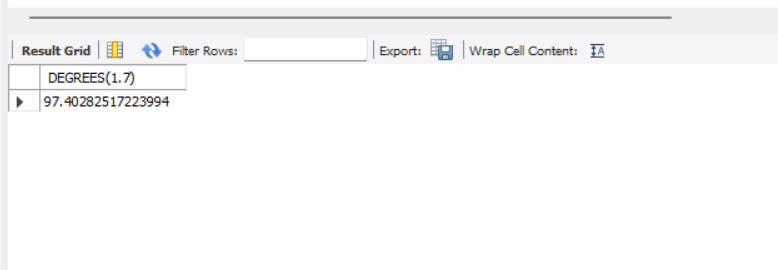
COUNT

SELECT COUNT(email) AS NumberOfEmails FROM Patient;



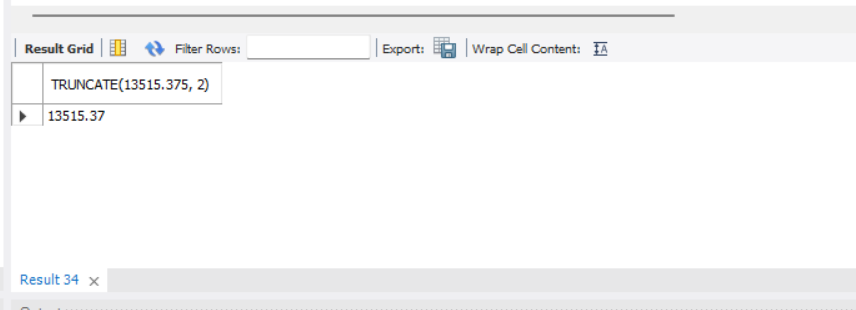
DEGREES

SELECT DEGREES(1.7);



TRUNCATE

SELECT TRUNCATE(13515.375, 2);



**Conclusion:**

Through this experiment, we learned about –

* String and Numeric Functions and how to implement them.
* Various String and Numeric Functions

**EXPERIMENT – 7**

**Aim:**

To study and write queries to implement joins.

**Theory:**

Aggregate

**SQL Join** statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are as follows:

* INNER JOIN
* LEFT JOIN
* RIGHT JOIN
* FULL JOIN

**INNER JOIN** keyword selects all rows from both the tables as long as the condition is satisfied. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be the same.

**LEFT JOIN** returns all the rows of the table on the left side of the join and matches rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result-set will contain *null*. LEFT JOIN is also known as LEFT OUTER JOIN.

**RIGHT JOIN** is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join. For the rows for which there is no matching row on the left side, the result-set will contain *null*. RIGHT JOIN is also known as RIGHT OUTER JOIN.

**FULL JOIN** creates the result-set by combining results of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both tables. For the rows for which there is no matching, the result-set will contain *NULL* values

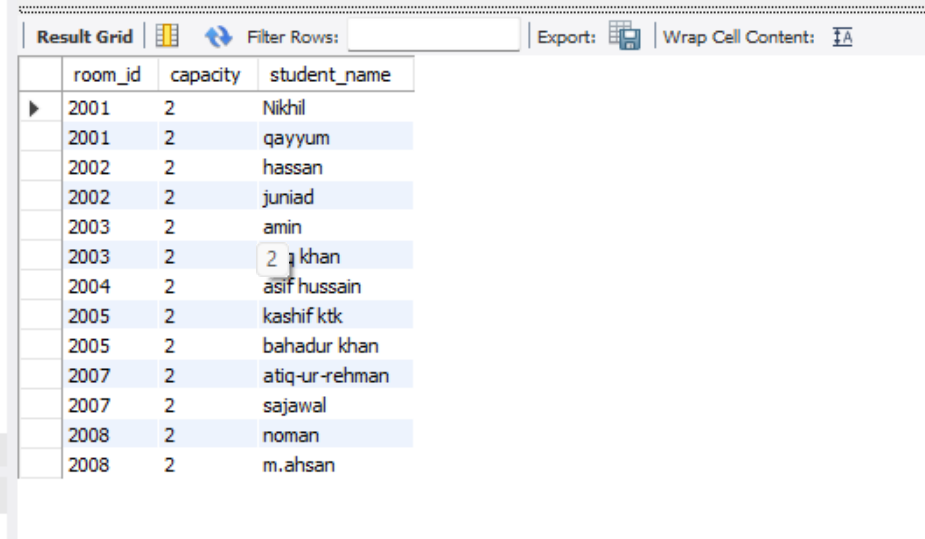
**Queries:**

INNER JOIN –

Select room.room\_id,room.capacity,student.student\_name from room

inner join student on room.room\_id=student.room\_id

where room.room\_status = "full";

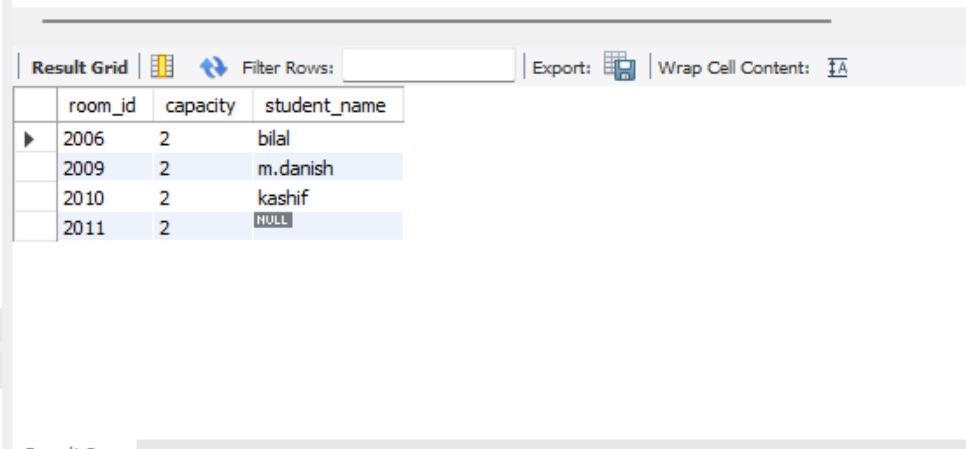


LEFT JOIN –

Select room.room\_id,room.capacity,student.student\_name from room

left join student on room.room\_id=student.room\_id

where room.room\_status = "not\_full";

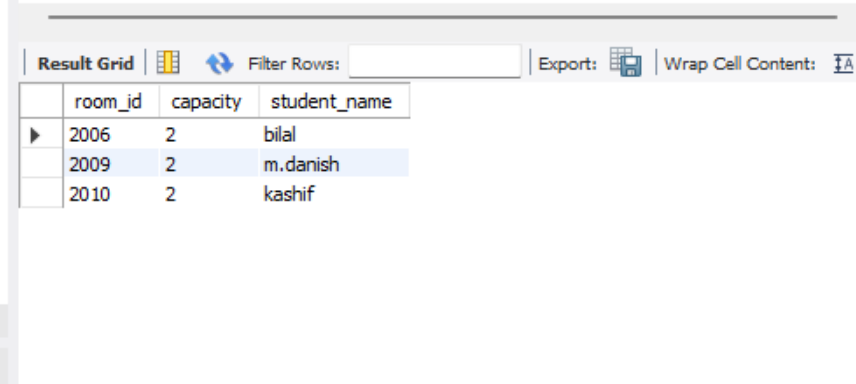


RIGHT JOIN –

Select room.room\_id,room.capacity,student.student\_name from room

right join student on room.room\_id=student.room\_id

where room.room\_status = "not\_full";

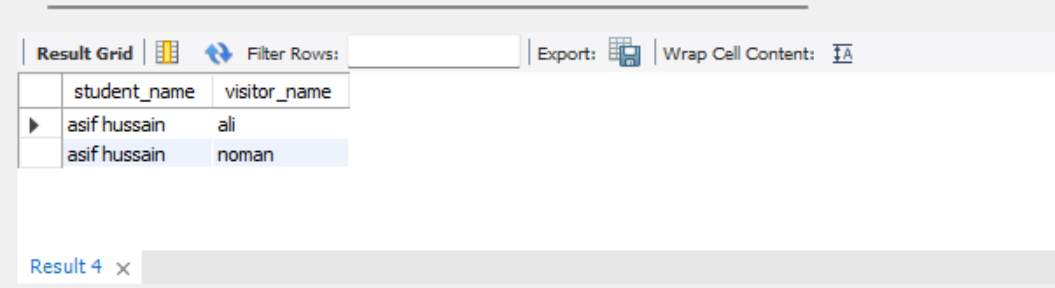


FULL JOIN –

Select student.student\_name,visitor.visitor\_name from student

full join visitor on student.student\_id=visitor.student\_id

where visitor.student\_id=107;



**Conclusions:**

Through this experiment, we learned about various joins such as INNER JOIN, LEFT JOIN, RIGHT JOIN and FULL JOIN

**EXPERIMENT – 8**

**Aim:** Write SQL Queries to Implement Correlated queries

**Theory:**

A correlated subquery is one way of reading every row in a table and comparing values in each row against related data. It is used whenever a subquery must return a different result or set of results for each candidate row considered by the main query. In other words, you can use a correlated subquery to answer a multipart question whose answer depends on the value in each row processed by the parent statement.

Nested Queries vs correlated Subqueries

With a normal nested subquery, the inner **SELECT** query runs first and executes once, returning values to be used by the main query. A correlated subquery, however, executes once for each candidate row considered by the outer query. In other words, the inner query is driven by the outer query.

**Queries:**

SELECT

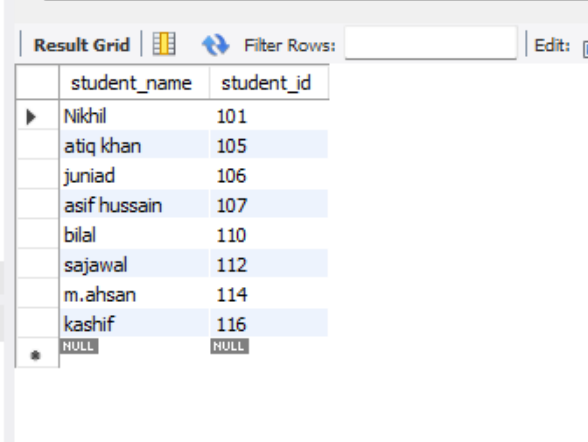
SELECT student\_name, student\_id FROM student

WHERE exists (

SELECT \* FROM visitor

WHERE visitor.student\_id = student.student\_id

);



UPDATE

DELETE

**Conclusion:**

Through this experiment, we learnt about correlated subqueries, how to write and difference between correlated subqueries and nested queries.

**EXPERIMENT – 9**

**Aim:** Write SQL Queries to implement WITH Clause

**Theory:**

The SQL WITH clause was introduced by Oracle in the Oracle 9i release 2 databases. The SQL WITH clause allows you to give a sub-query block a name (a process also called sub-query refactoring), which can be referenced in several places within the main SQL query.

* The clause is used for defining a temporary relation such that the output of this temporary relation is available and is used by the query that is associated with the WITH clause.
* Queries that have an associated WITH clause can also be written using nested sub-queries but doing so add more complexity to read/debug the SQL query.
* WITH clause is not supported by all database system.
* The name assigned to the sub-query is treated as though it was an inline view or table
* The SQL WITH clause was introduced by Oracle in the Oracle 9i release 2 databases.

**Queries:**

1. Students having more than average age

WITH temporaryTable (averageValue) as

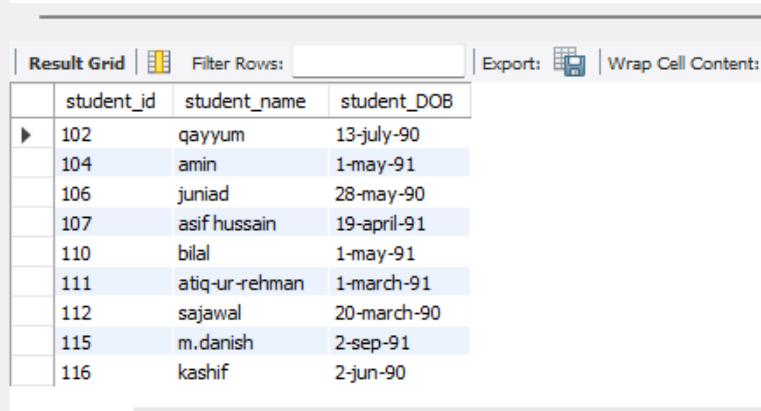
(SELECT avg(student\_age)

FROM student)

SELECT student\_id,student\_name,student\_DOB

FROM student, temporaryTable

WHERE student.student\_age > temporaryTable.averageValue



1. Hostels having number of students more than 50 with more than average annual expenses

WITH temporaryTable (averageValue) as

(SELECT avg(annual\_expences)

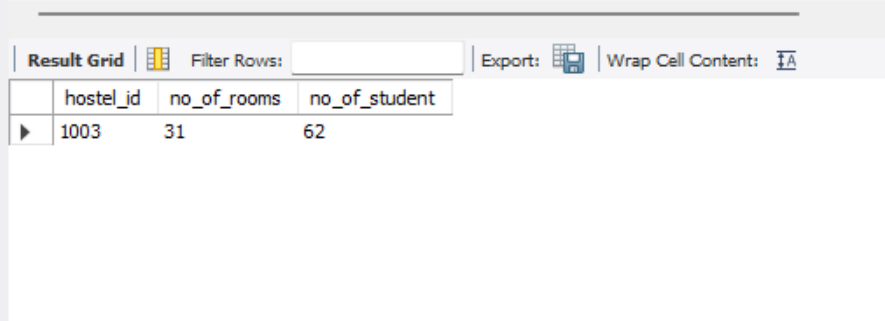
FROM boys\_hostel

where no\_of\_student > 50)

SELECT hostel\_id,no\_of\_rooms,no\_of\_student

FROM boys\_hostel, temporaryTable

WHERE boys\_hostel.annual\_expences > temporaryTable.averageValue



**Conclusion:**

Through this experiment, we learned about use of WITH clause. We also learned how temporary table can be used for specific selections in database without the need to create separate queries.

**EXPERIMENT – 10**

**Aim:** Write a program to study PL/SQL

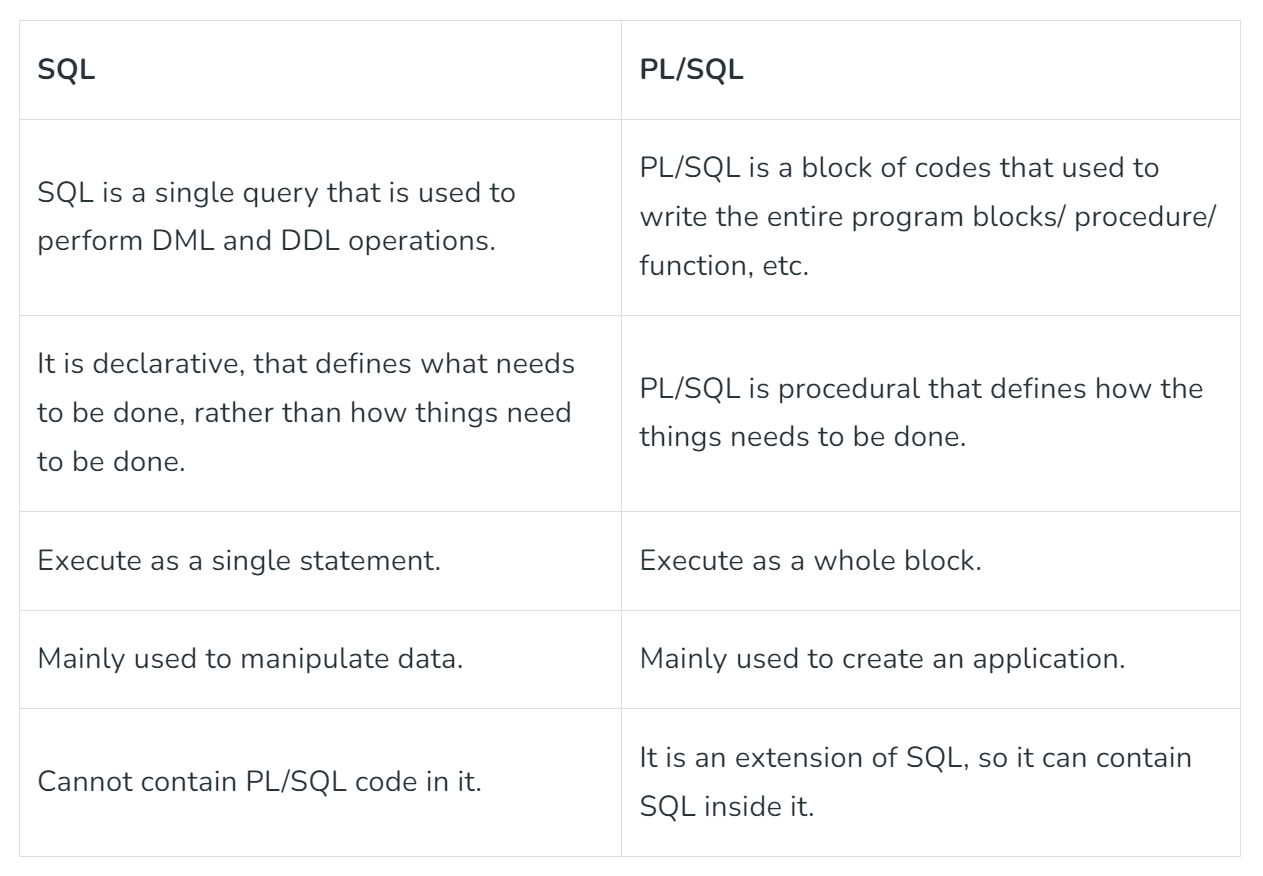
**Theory:**

PL/SQL is a block structured language that enables developers to combine the power of SQL with procedural statements. All the statements of a block are passed to oracle engine all at once which increases processing speed and decreases the traffic.

**Basic Of PL/SQL -**

* PL/SQL stands for Procedural Language extensions to the Structured Query Language (SQL).
* PL/SQL is a combination of SQL along with the procedural features of programming languages.
* Oracle uses a PL/SQL engine to processes the PL/SQL statements.
* PL/SQL includes procedural language elements like conditions and loops. It allows declaration of constants and variables, procedures and functions, types and variable of those types and triggers.

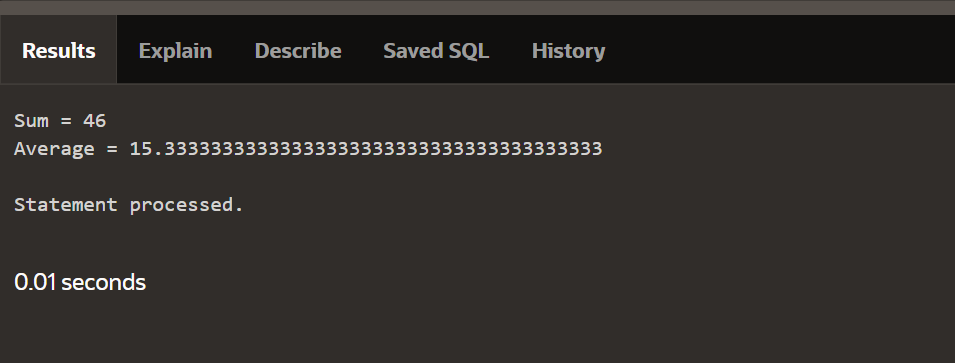
**SQL v/s PL/SQL –**



**Queries:**

Program to Find Sum and Avg of three numbers

|  |
| --- |
| --To find sum and avg of three numbers  **DECLARE**      -- Assigning 12 into a      a     NUMBER := 12;        -- Assigning 14 into b      b     NUMBER := 14;        -- Assigning 20 into c      c     NUMBER := 20;        -- Declare variable for sum Of Three number (a, b, c)      sumOf3 NUMBER;        -- Declare variable for average      avgOf3 NUMBER;    --Start Block  **BEGIN**      -- Assigning sum of a, b and c into sumOf3      sumOf3 := a + b + c;        -- Assigning average  of a, b and c into avgOf3      avgOf3 := sumOf3 / 3;        --print Result  sum of a, b, c number      dbms\_output.Put\_line('Sum = '||sumOf3);        --print Average  sum of a, b, c number      dbms\_output.Put\_line('Average = '||avgOf3);  **END**; |

****

**Conclusion:**

* In this experiment, we studied about the PL/SQL language.
* We learnt that we can combine various programming constructs like loops and conditional statements in the basic SQL using PL/SQL.
* The various constructs that we learnt about in this experiment are DECLARE, BEGIN, END, IF statements, FOR loop, WHILE loop, Functions and Procedures.