**Chandigarh Group of Colleges**

**College of Engineering, Landran, Mohali-140307**

Department of Computer Science & Engineering



**Database Management System Lab File**

**(BTCS-501-18)**

SUBMITTED TO SUBMITTED BY   
Ms. Jagjit Kaur Abhinandan  
 1914565

CSE

5th Semester

Group – Z

**INDEX**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **PRACTICAL** | **PAGE NO.** |
| 1 | Introduction to SQL | 2-4 |
| 2 | Installation of SQL Server / Oracle | 5-9 |
| 3 | To Create, Alter and Drop Tables in Oracle Database | 10-12 |
| 4 | To Insert, Select and Update in Oracle Database | 13-14 |
| 5 | TO MAKE USE OF DIFFERENT CLAUSES VIZ WHERE , GROUP BY , HAVING , ORDER BY , UNION , INTERSECTION , SET DIFFERENCE | 15-19 |
| 6 | To perform various arithmetic operators and logical operators on database | 20-22 |
| 7 | To Study Differnet Constraints. | 23-27 |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |
| 11 |  |  |

**Practical:-01**

**Aim :-** Introduction to SQL and installation of SQL Server / Oracle

**Introduction :-** Structured Query Language (SQL) is the set of statements with which all programs and users access data in an Oracle database. Application programs and Oracle tools often allow users access to the database without using SQL directly, but these applications in turn must use SQL when executing the user's request. This chapter provides background information on SQL as used by most database systems.

* **What is SQL?**

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.

SQL is the standard language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

Also, they are using different dialects, such as −

* MS SQL Server using T-SQL,
* Oracle using PL/SQL,
* MS Access version of SQL is called JET SQL (native format) etc.
* **Why SQL?**

SQL is widely popular because it offers the following advantages −

* Allows users to access data in the relational database management systems.
* Allows users to describe the data.
* Allows users to define the data in a database and manipulate that data.
* Allows to embed within other languages using SQL modules, libraries & pre-compilers.
* Allows users to create and drop databases and tables.
* Allows users to create view, stored procedure, functions in a database.
* Allows users to set permissions on tables, procedures and views.
* **A Brief History of SQL**
* **1970** − Dr. Edgar F. "Ted" Codd of IBM is known as the father of relational databases. He described a relational model for databases.
* **1974** − Structured Query Language appeared.
* **1978** − IBM worked to develop Codd's ideas and released a product named System/R.
* **1986** − IBM developed the first prototype of relational database and standardized by ANSI. The first relational database was released by Relational Software which later came to be known as Oracle.
* **SQL Process**

When you are executing an SQL command for any RDBMS, the system determines the best way to carry out your request and SQL engine figures out how to interpret the task.

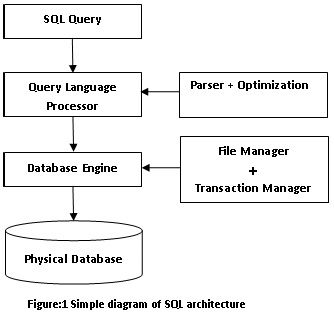
There are various components included in this process.

These components are −

* Query Dispatcher
* Optimization Engines
* Classic Query Engine
* SQL Query Engine, etc.

A classic query engine handles all the non-SQL queries, but a SQL query engine won't handle logical files.

Following is a simple diagram showing the SQL Architecture −



* **SQL Commands**

The standard SQL commands to interact with relational databases are CREATE, SELECT, INSERT, UPDATE, DELETE and DROP. These commands can be classified into the following groups based on their nature –

* **DDL - Data Definition Language**

|  |  |
| --- | --- |
| **Sr.No.** | **Command & Description** |
| 1 | **CREATE**  Creates a new table, a view of a table, or other object in the database. |
| 2 | **ALTER**  Modifies an existing database object, such as a table. |
| 3 | **DROP**  Deletes an entire table, a view of a table or other objects in the database. |

* **DML - Data Manipulation Language**

|  |  |
| --- | --- |
| **Sr.No.** | **Command & Description** |
| 1 | **SELECT**  Retrieves certain records from one or more tables. |
| 2 | **INSERT**  Creates a record. |
| 3 | **UPDATE**  Modifies records. |
| 4 | **DELETE**  Deletes records. |

* **DCL - Data Control Language**

|  |  |
| --- | --- |
| **Sr.No.** | **Command & Description** |
| 1 | **GRANT**  Gives a privilege to user. |
| 2 | **REVOKE**  Takes back privileges granted from user. |

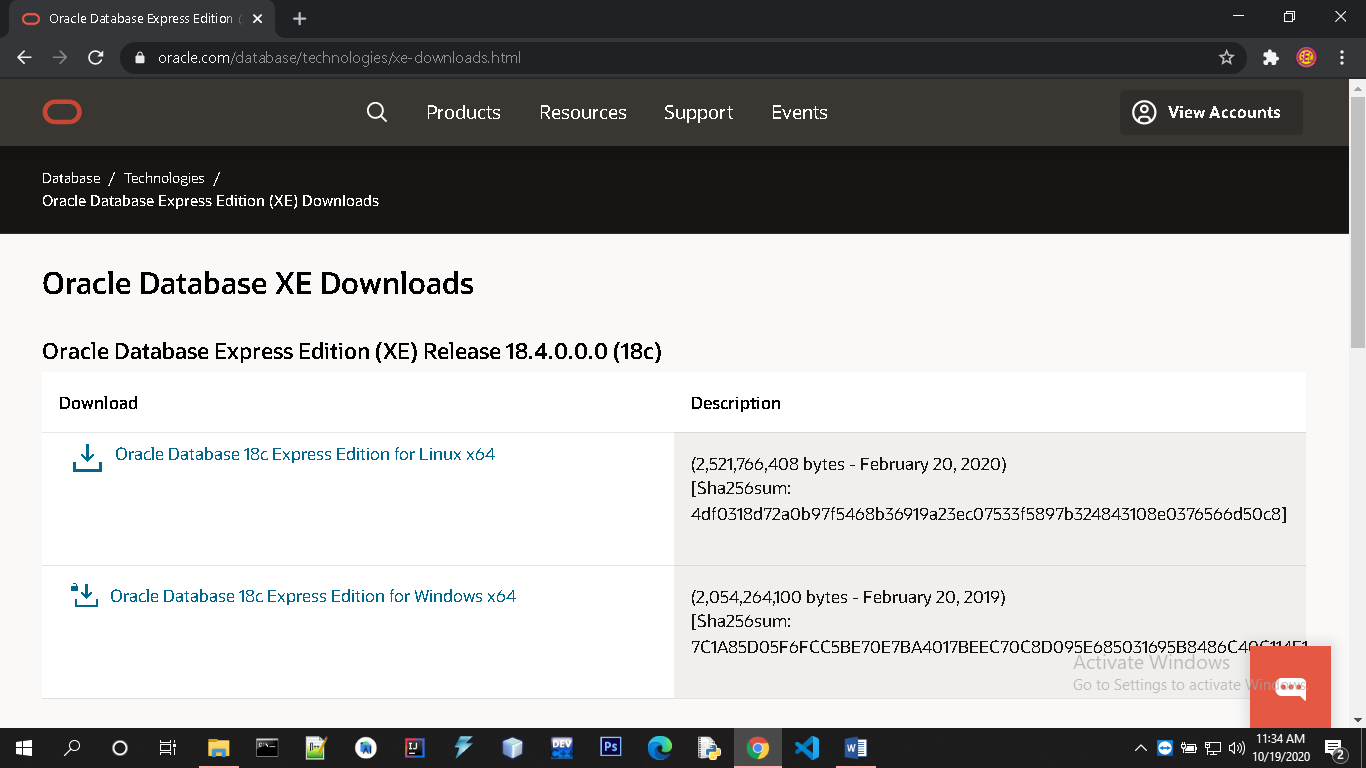
**Practical:-02**

**Aim :-** Installation of SQL Server / Oracle

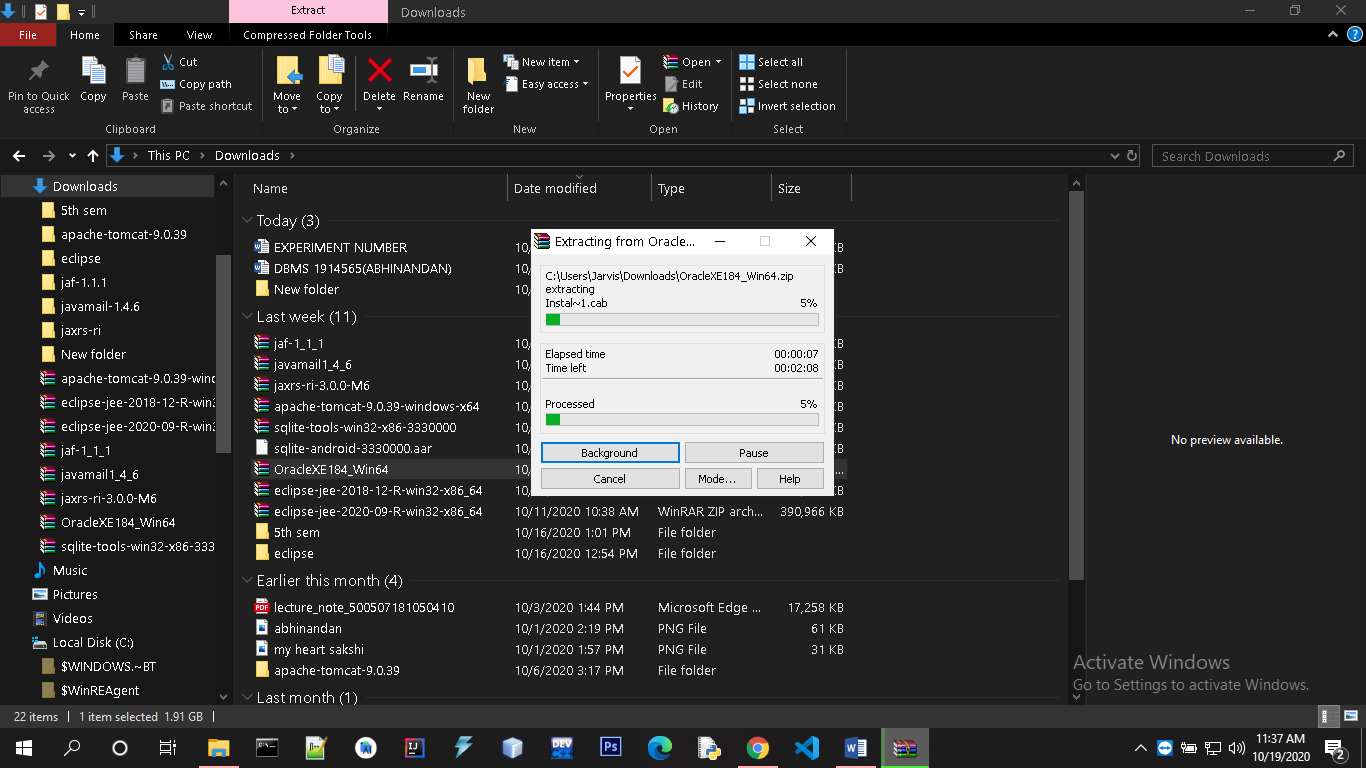
* **Install Oracle 18c Express Edition**

Following are the steps involved in installing the Oracle 18c Express Edition in Window Server:

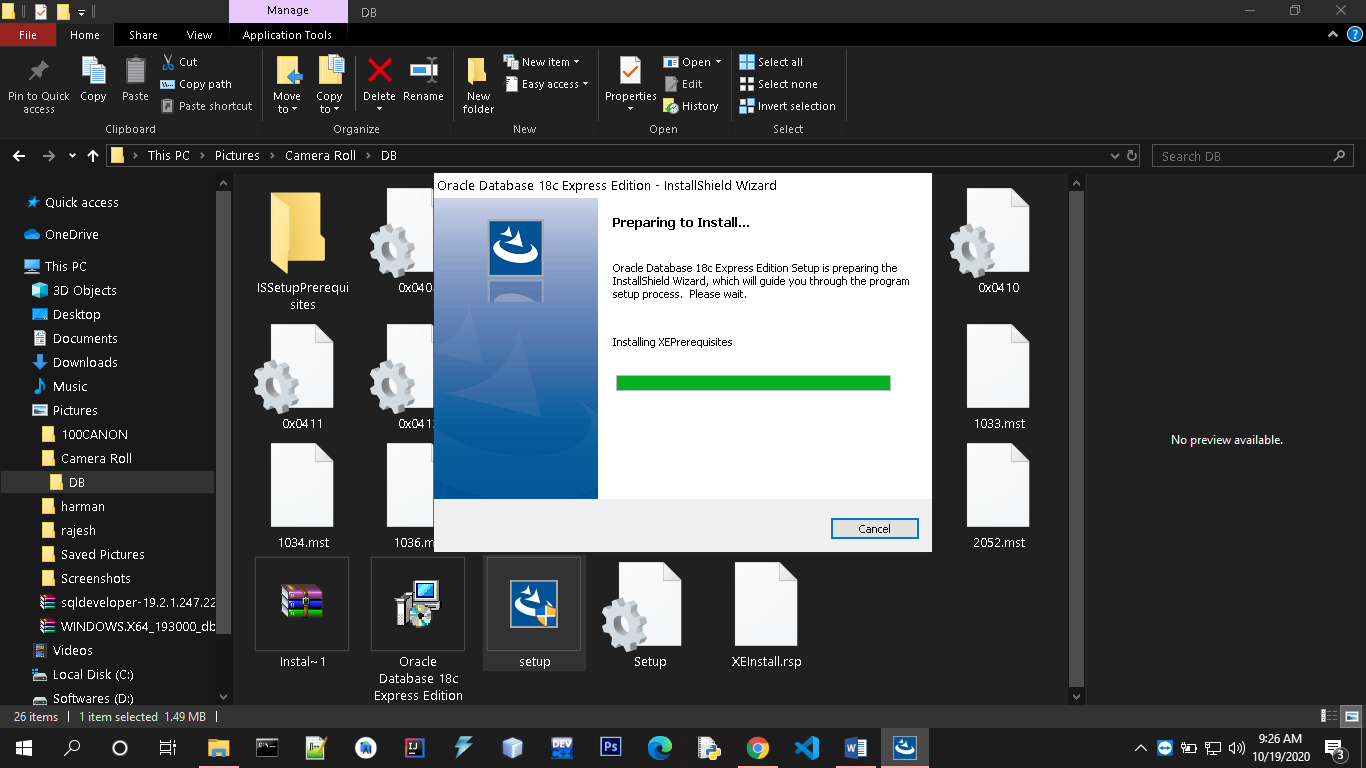
**1. Download the Oracle 18c Express edition software from Oracle Site.**



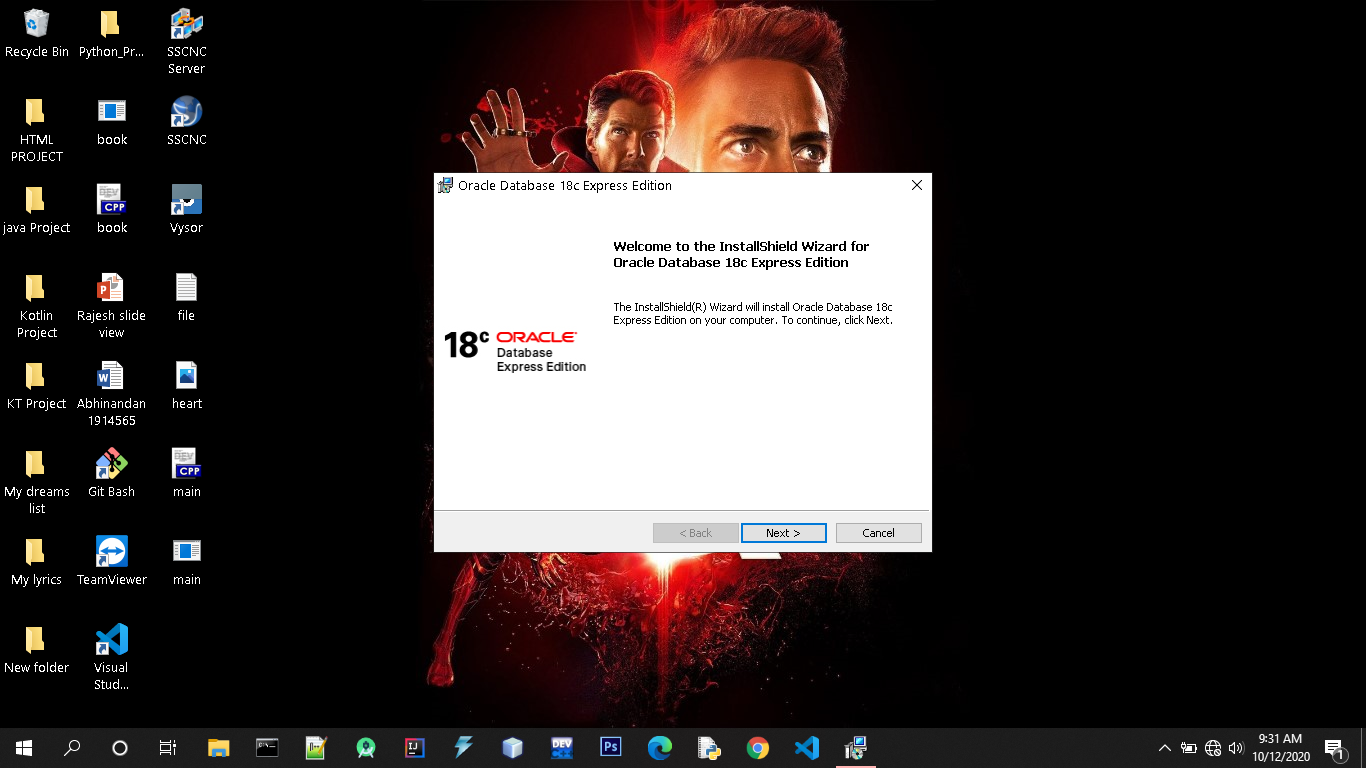
**2. Unzip the Software to folder.**



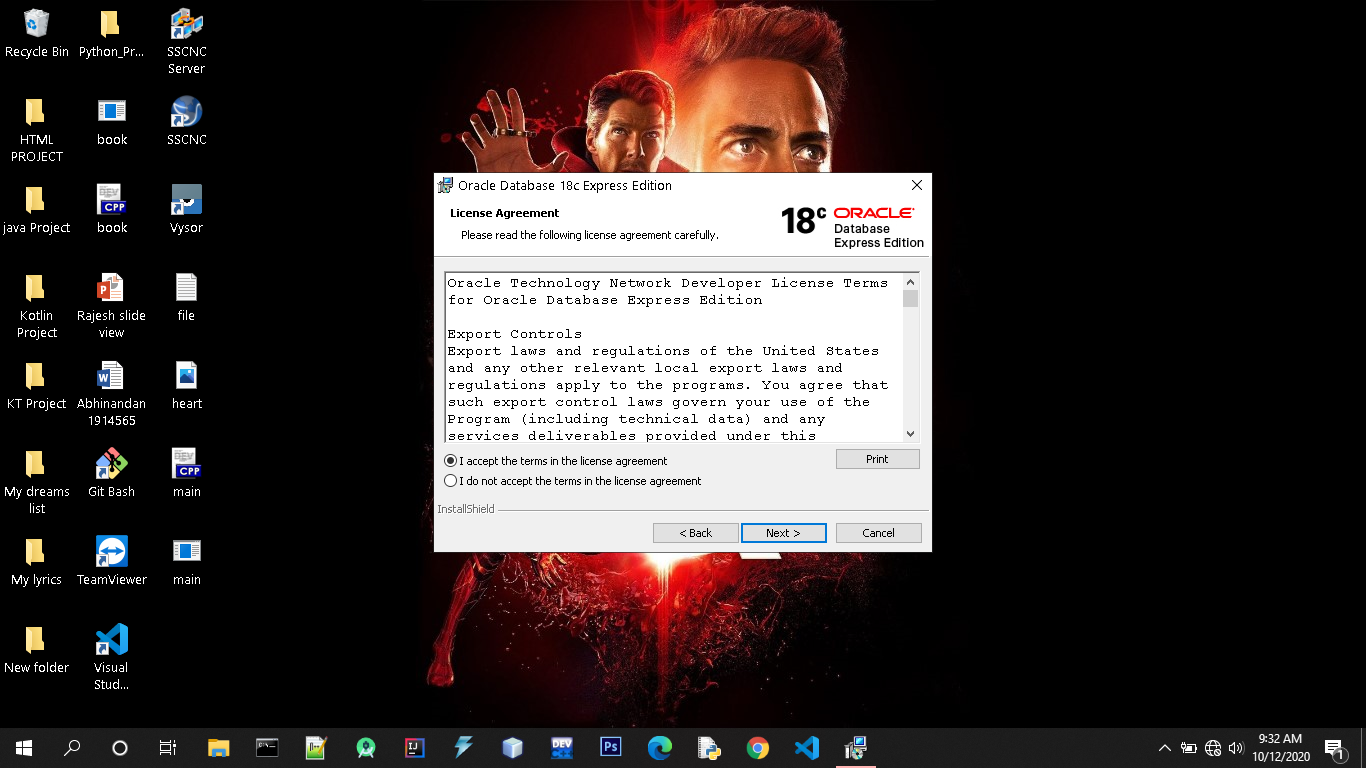
**3. Run the Setup.exe utility and get the following screen of prerequisites.**

****

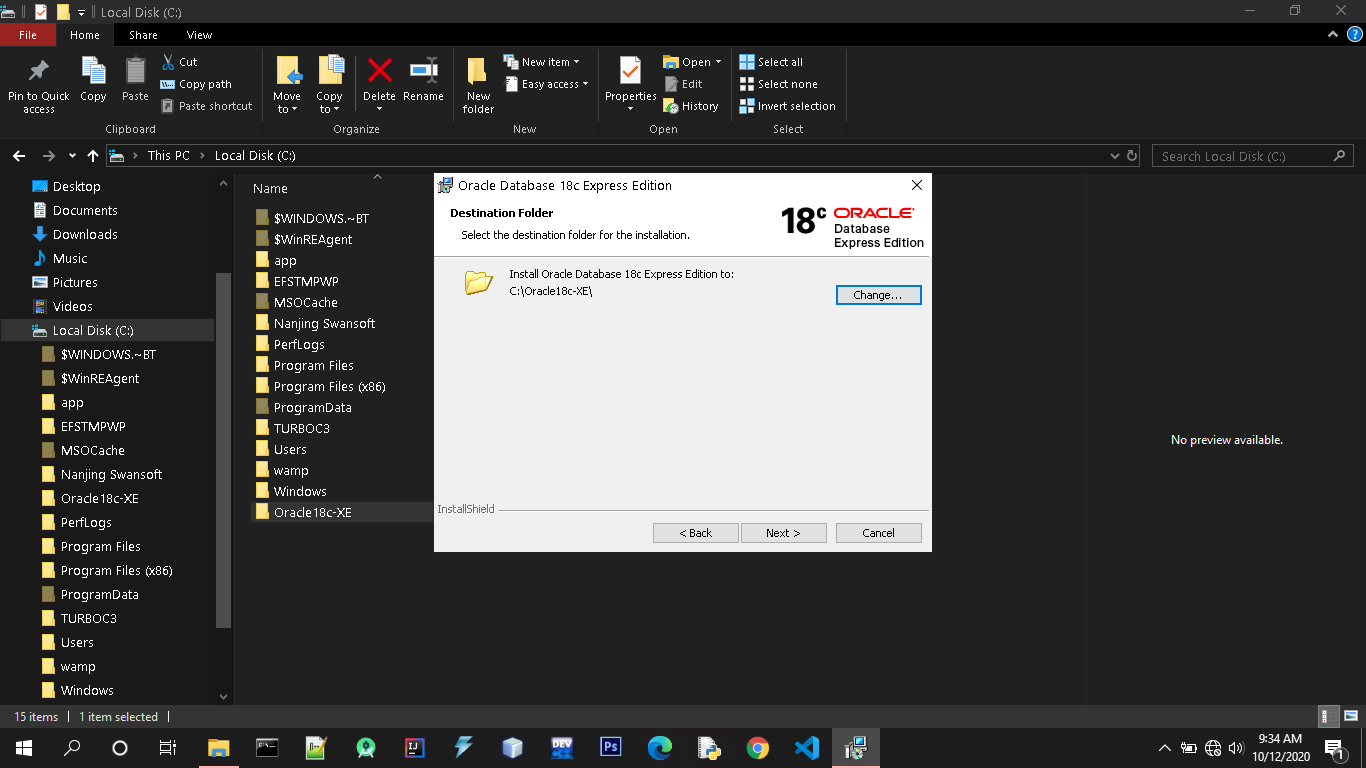
**4. After prerequisites you will start the installation process by pressing NEXT.**

****

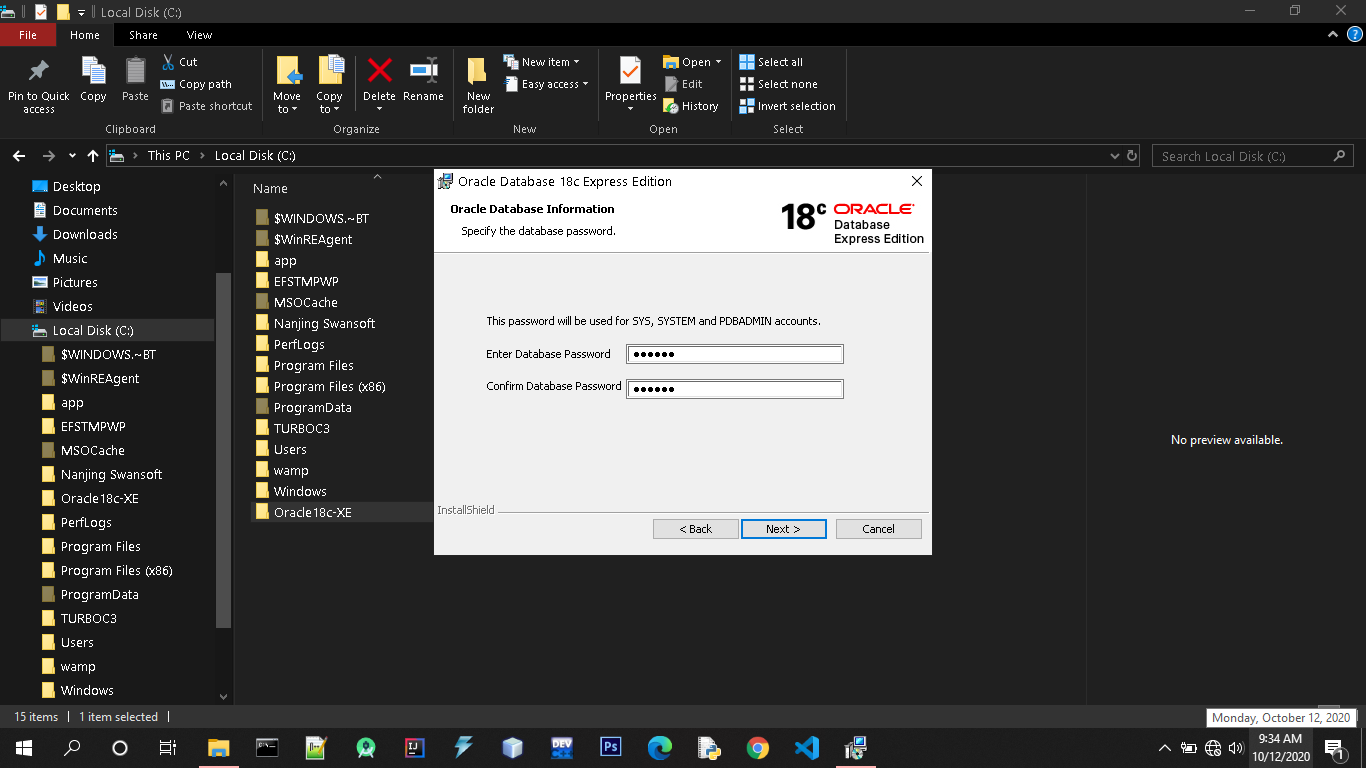
**5. You should accept the license agreement.**

****

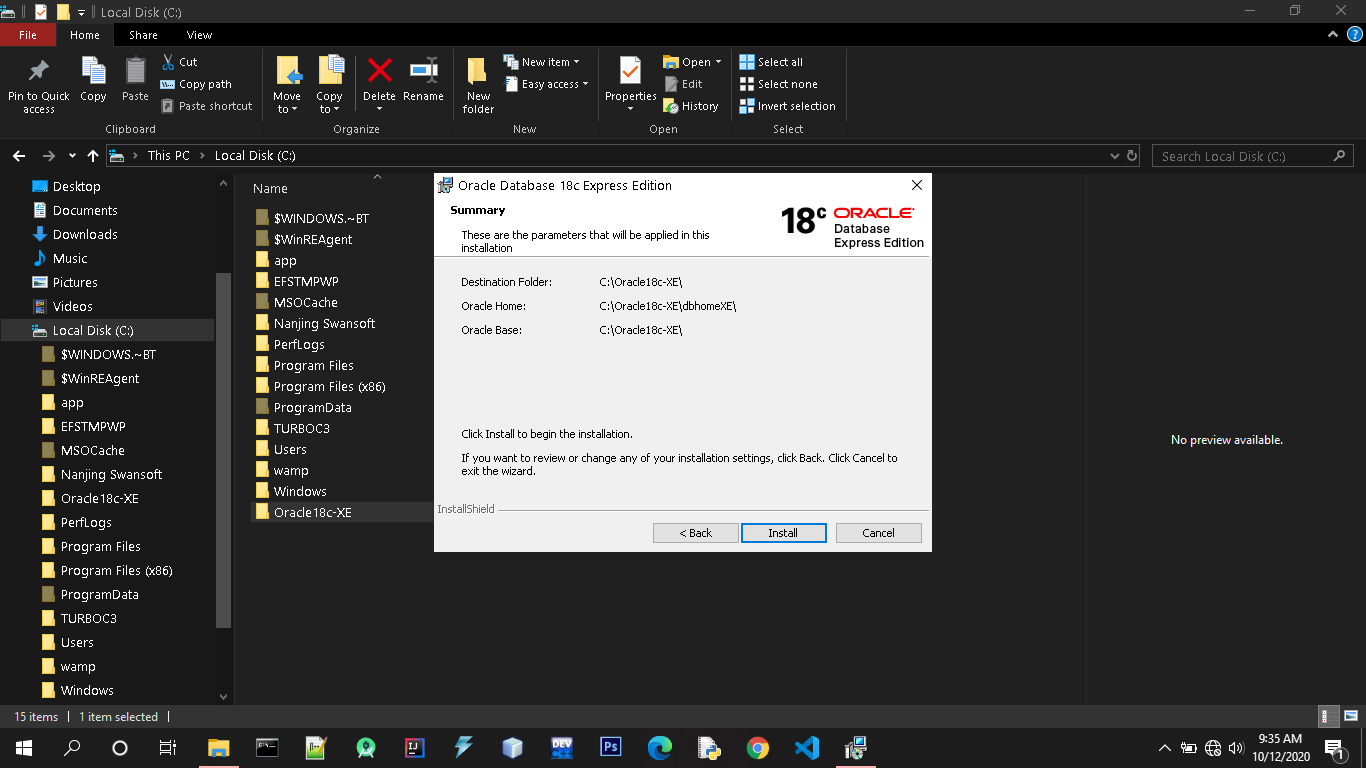
**6. Choose the Oracle Base and Home location.**

****

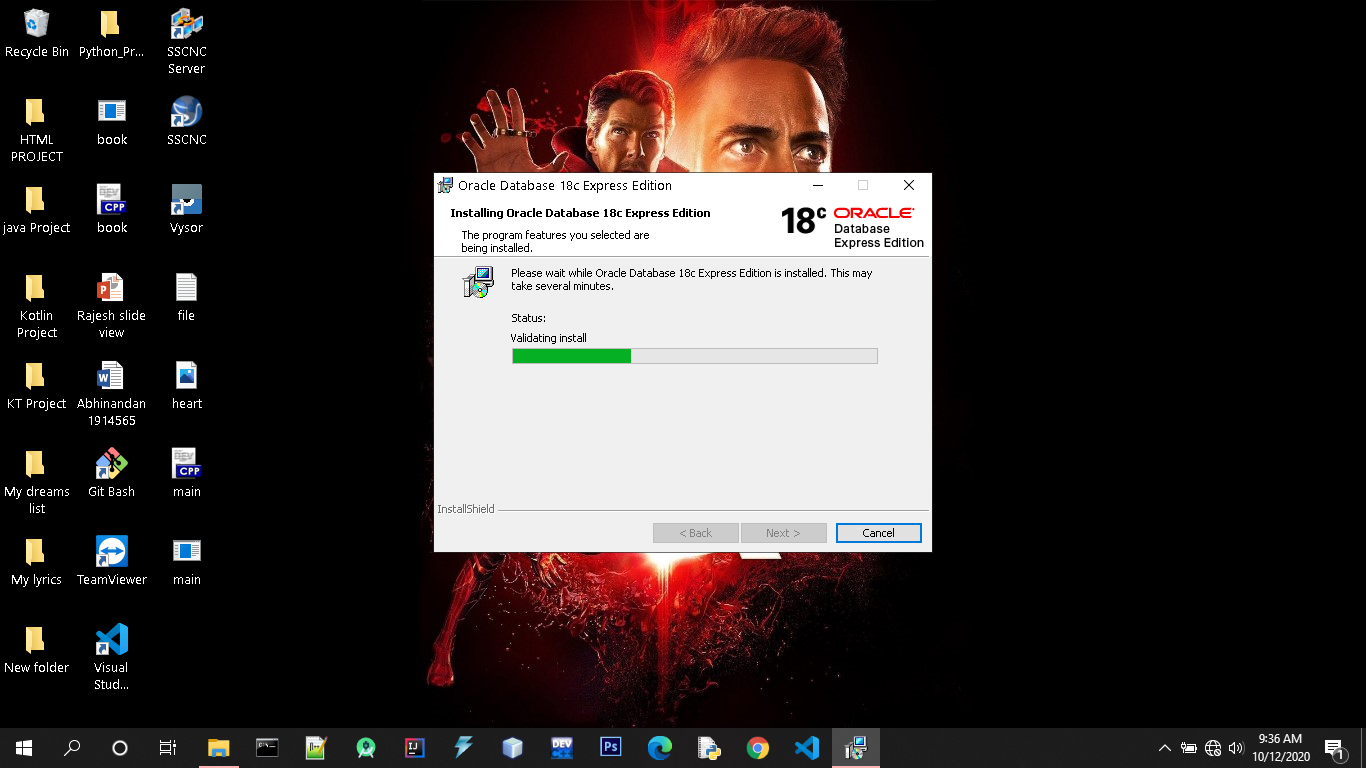
**7. Set the Password for sys and system user.**

****

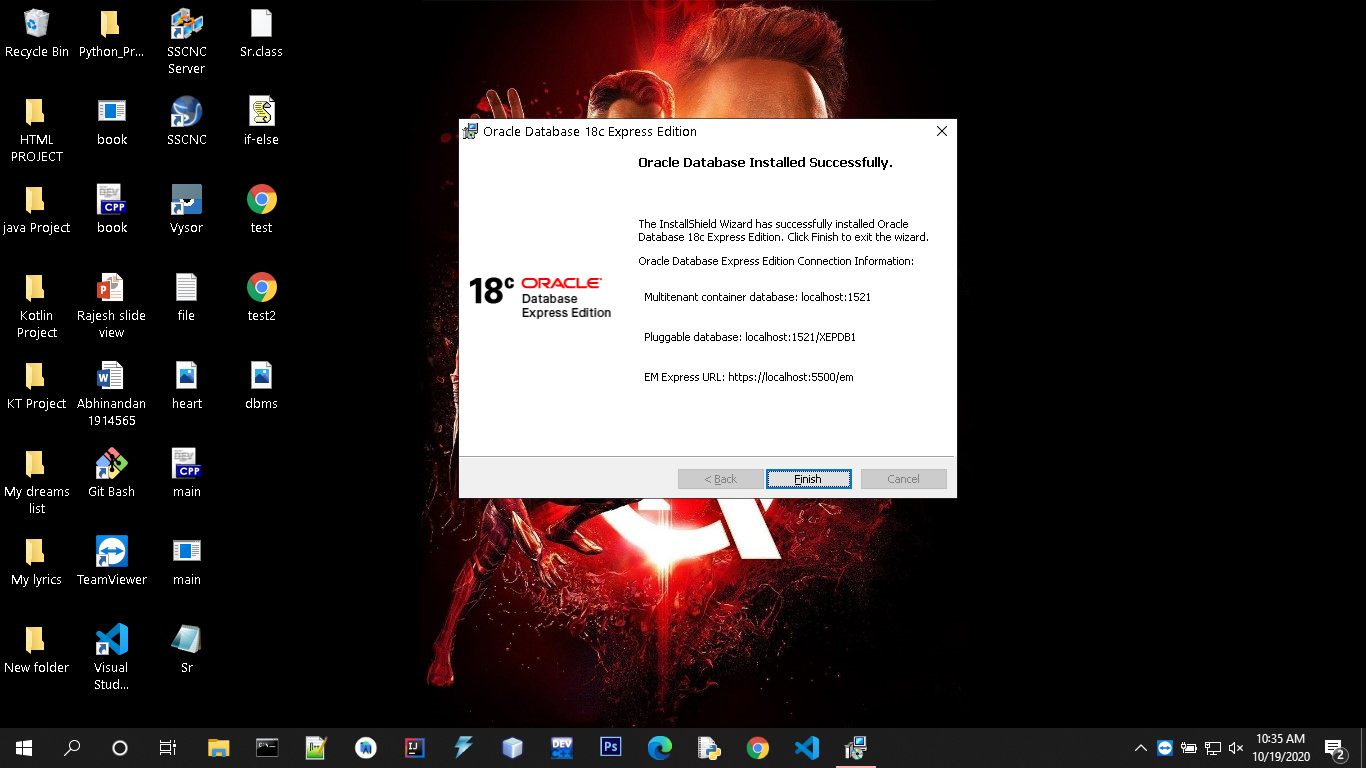
**8. Verify the destination where oracle is going to install and press NEXT.**

****

**9. Start the Installation of Oracle 18c Express Edition.**

****

**10. Finish the installation process.**

****

**Practical:-03**

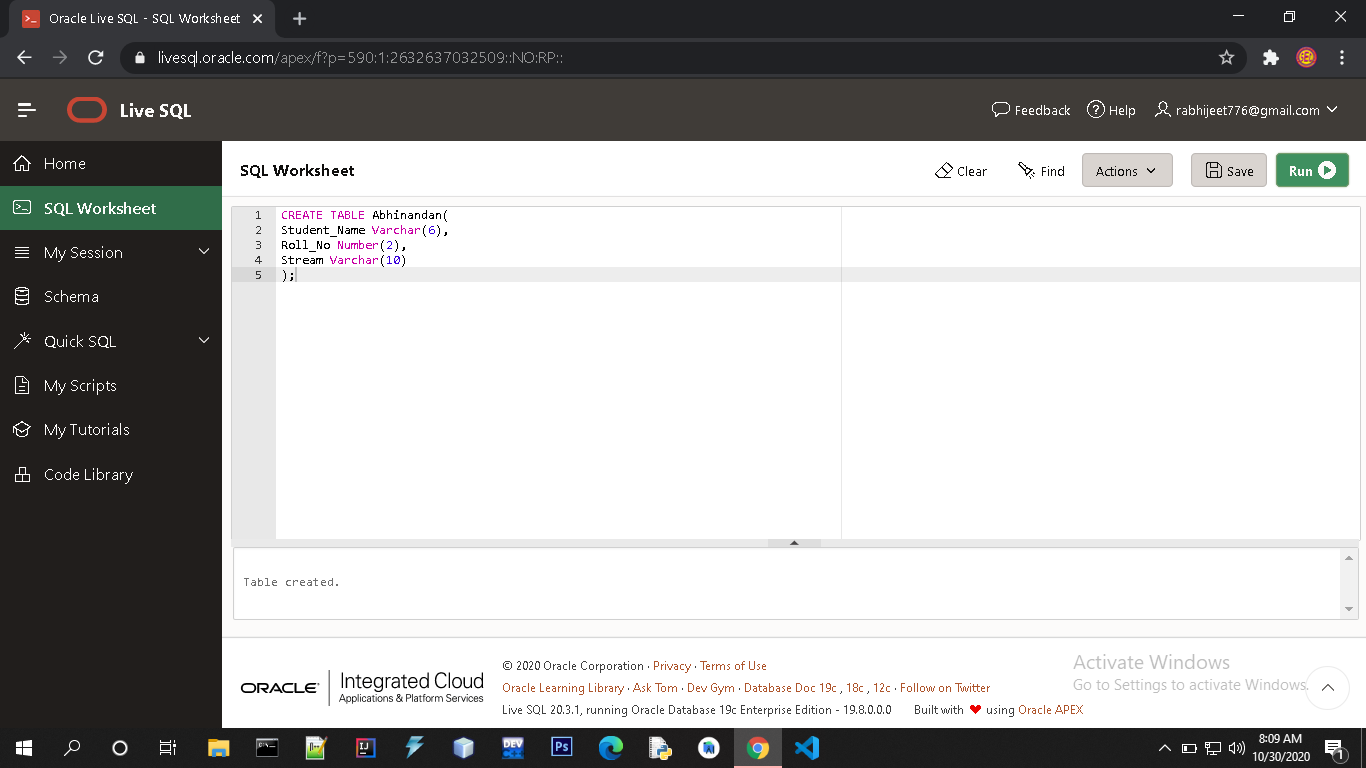
Aim: To Create, Alter and Drop Tables in Oracle Database

Create Table Statement:

The CREATE TABLE statement is used to create a new table in a database.

Syntax-

CREATE TABLE table\_name(column1 datatype*,* column2 datatype,column3 datatype, ....);



Alter Table Statement:

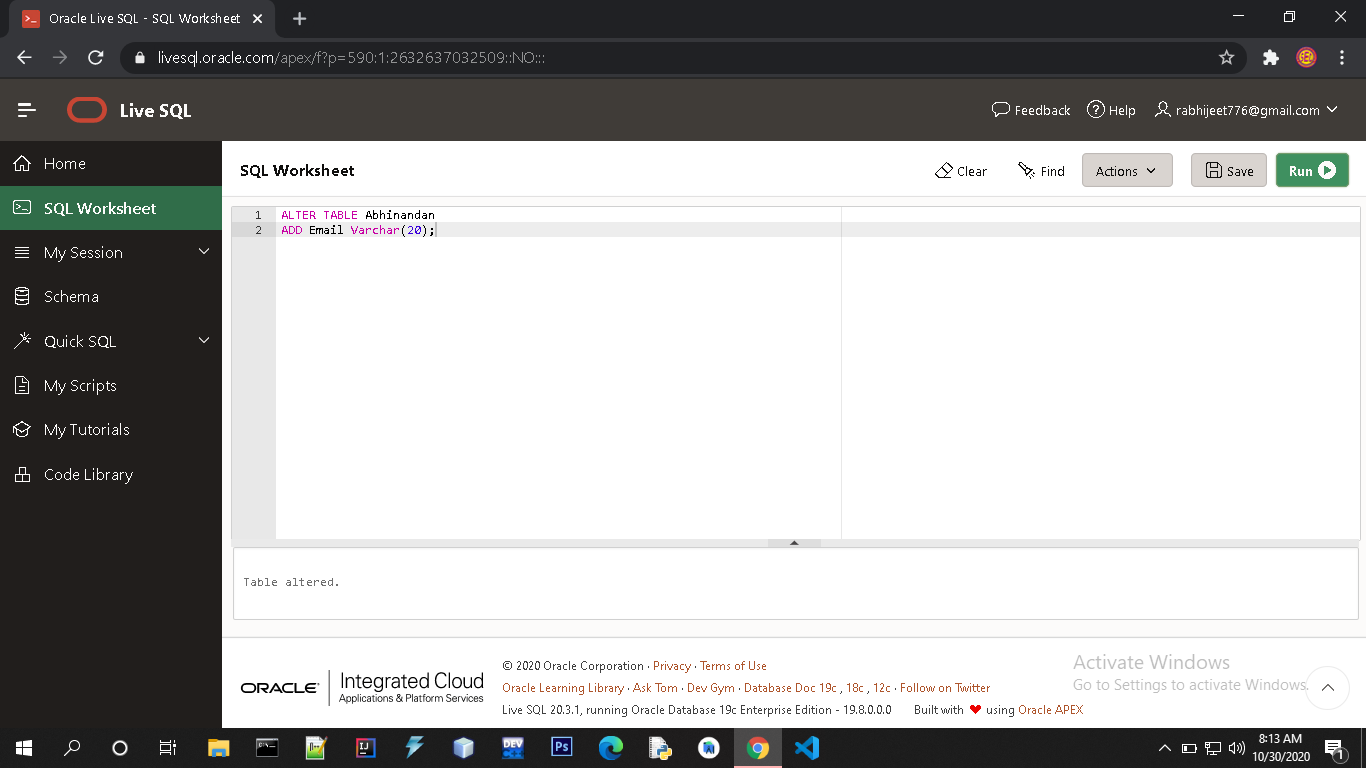
The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.

The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

=>ALTER TABLE - ADD Column

To add a column in a table, use the following syntax:

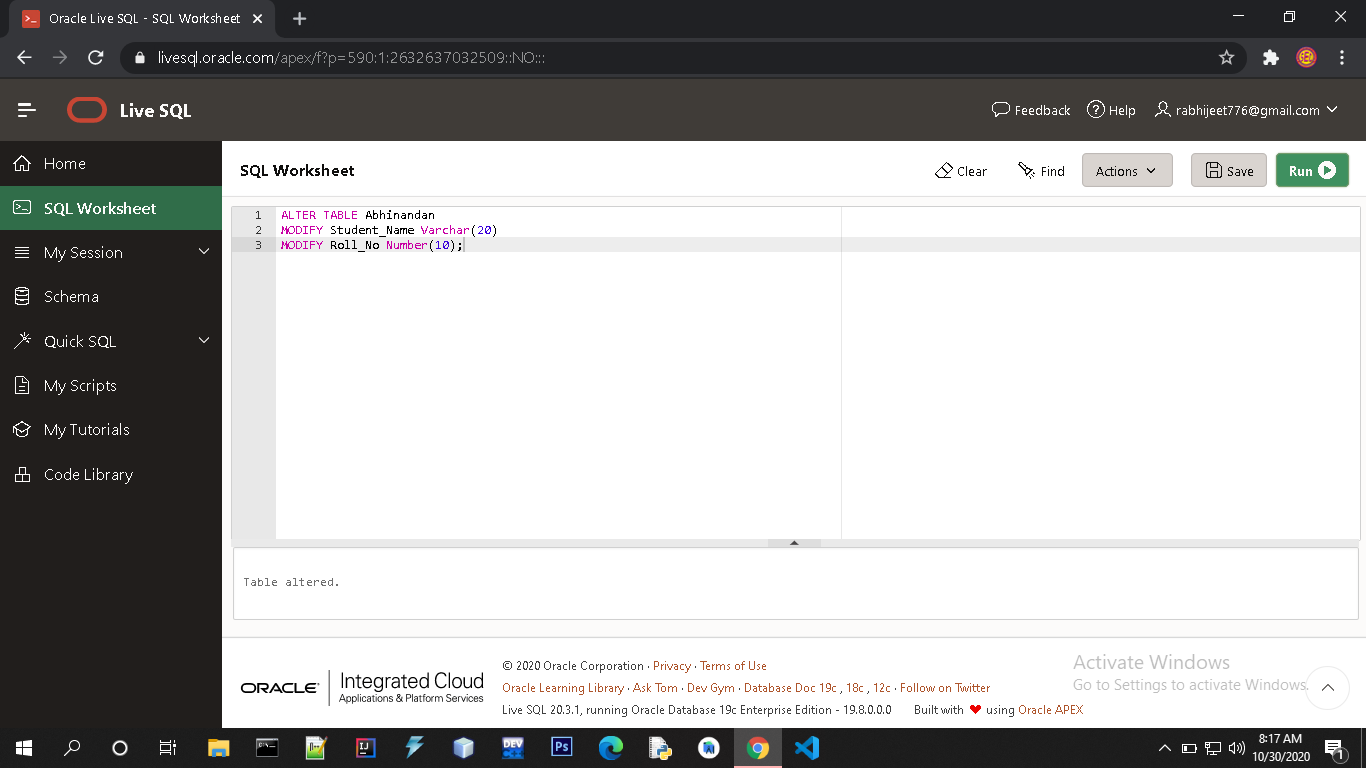
ALTER TABLE table\_name  
ADD column\_name datatype;



=>ALTER TABLE - MODIFY COLUMN

To change the data type of a column in a table, use the following syntax:

ALTER TABLE table\_name  
MODIFY COLUMN column\_name datatype;

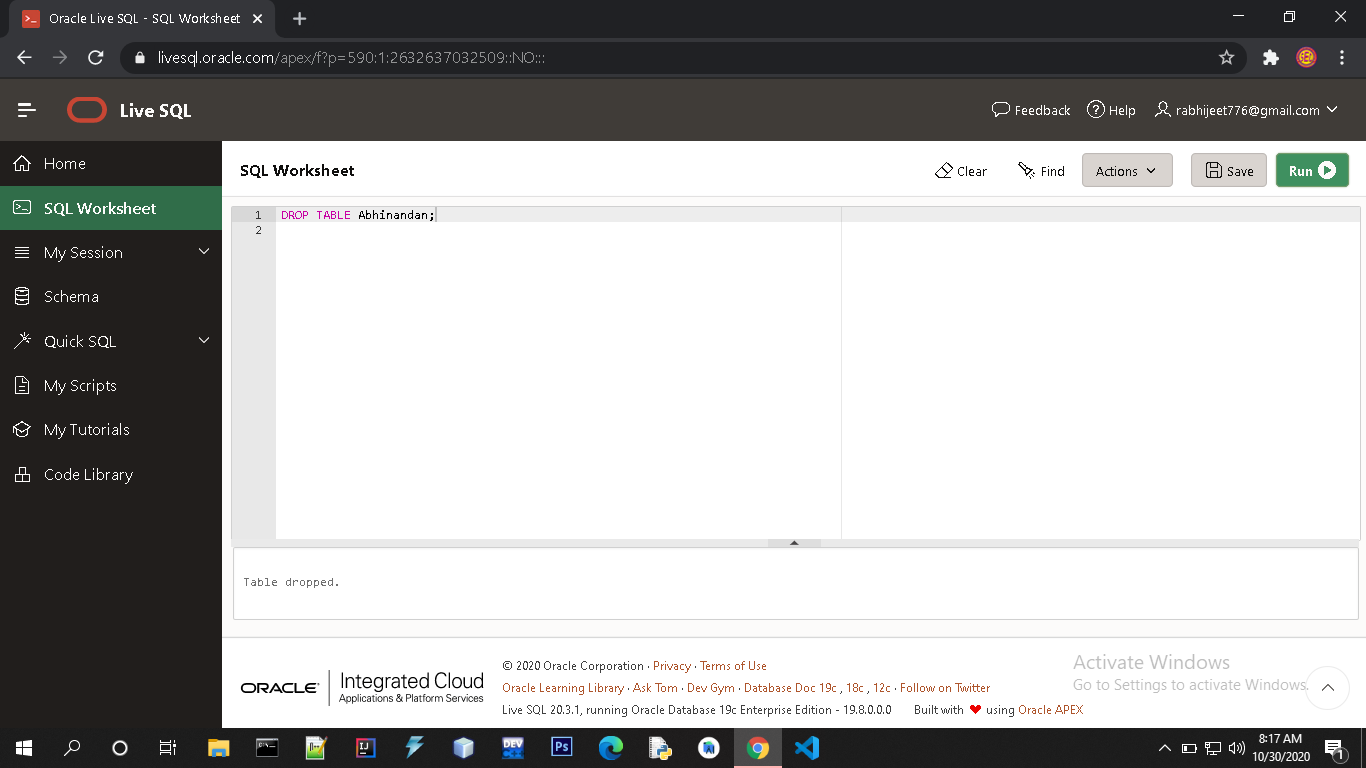


Drop Table Statement:

The DROP TABLE statement is used to drop an existing table in a database.

Syntax-

DROP TABLE table\_name;



**EXPERIMENT No:4**

Aim: To Insert, Select and Update in Oracle Database

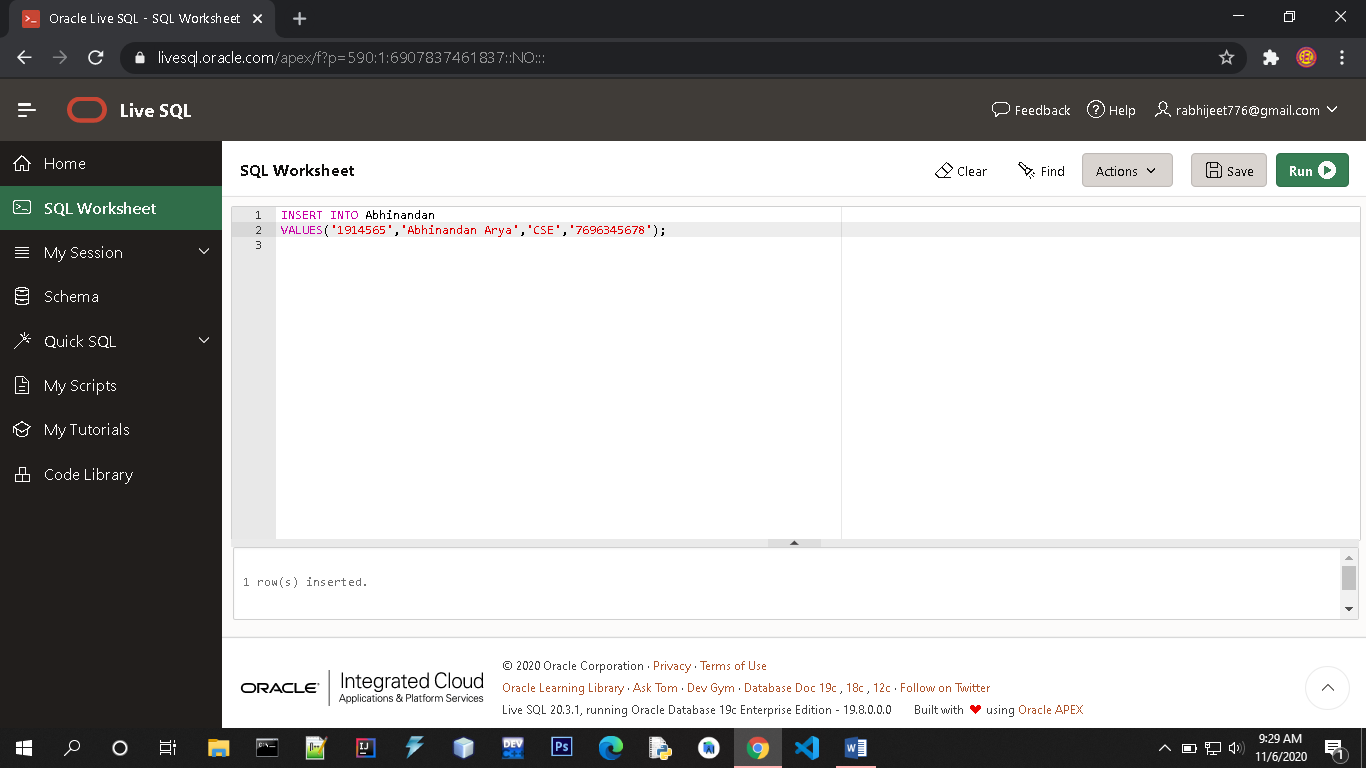
INSERT INTO Statement:

The INSERT INTO statement is used to insert new records in a table.

The first way specifies both the column names and the values to be inserted:-

Syntax-

INSERT INTO table\_name (column1, column2, column3, ...)  
VALUES (value1, value2, value3, ...);



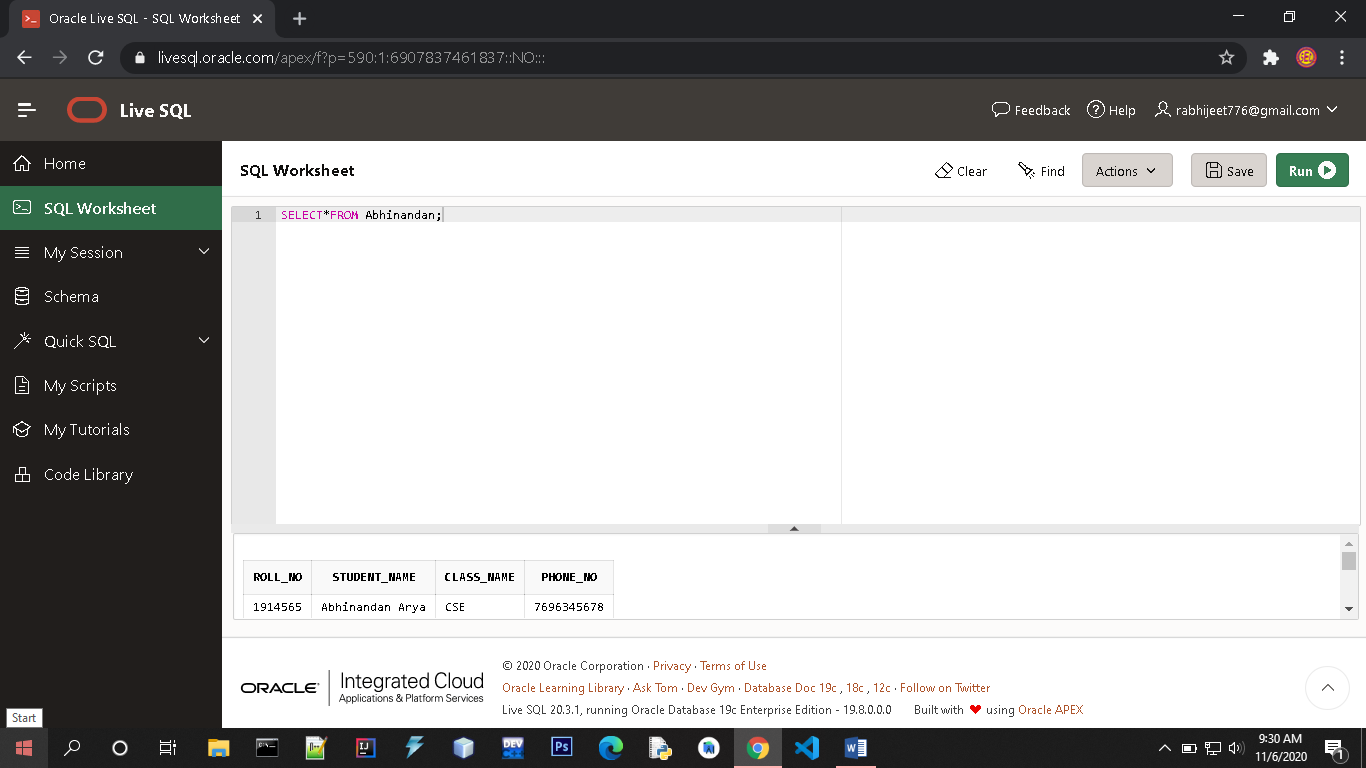
SELECT Statement:

The SELECT statement is used to select data from a database.

The data returned is stored in a result table, called the result-set.

Syntax-

SELECT \* FROM table\_name;

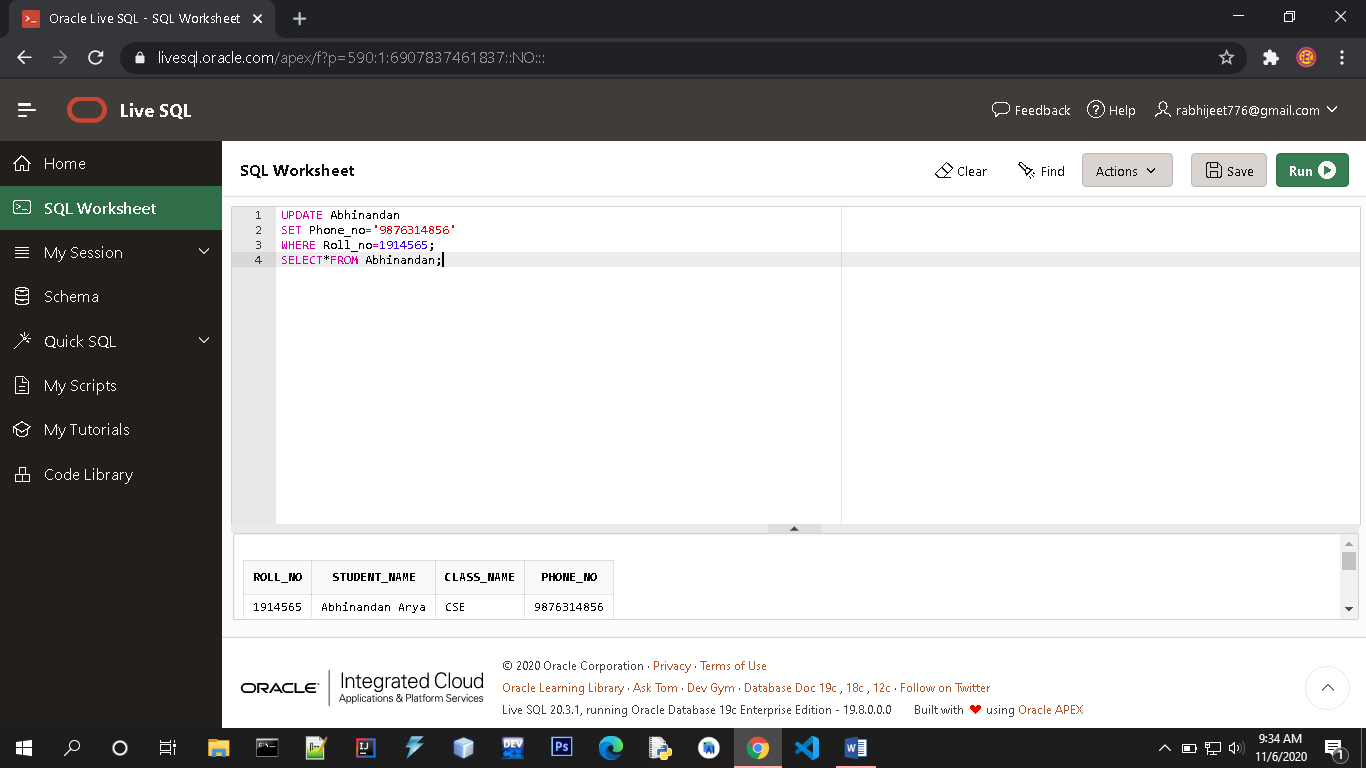


UPDATE Statement:

The UPDATE statement is used to modify the existing records in a table.

Syntax-

UPDATE table\_name  
SET column1 *=* ‘value1’, column2= ‘value2’, …  
WHERE condition;



**PRACTICAL :- 05**

**Aim :- TO MAKE USE OF DIFFERENT CLAUSES VIZ WHERE , GROUP BY , HAVING , ORDER BY , UNION , INTERSECTION , SET DIFFERENCE**

**A) WHERE CLAUSE**

The WHERE clause is used to extract only those records that fulfill a specified condition.

SYNTAX: SELECT *column1*, *column2, ...*  
 FROM *table\_name*  
 WHERE *condition*;

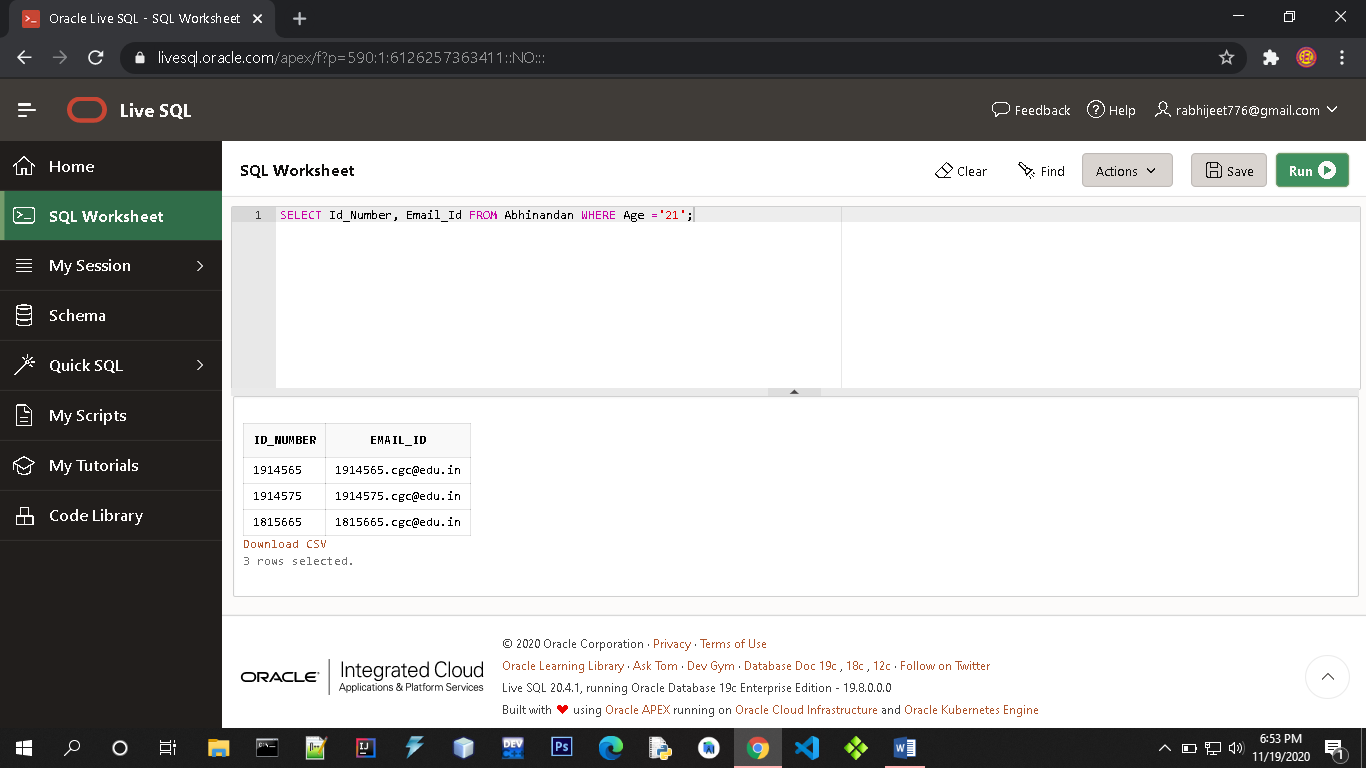
**Operators in the WHERE Clause**

The following operators can be used in the WHERE clause:

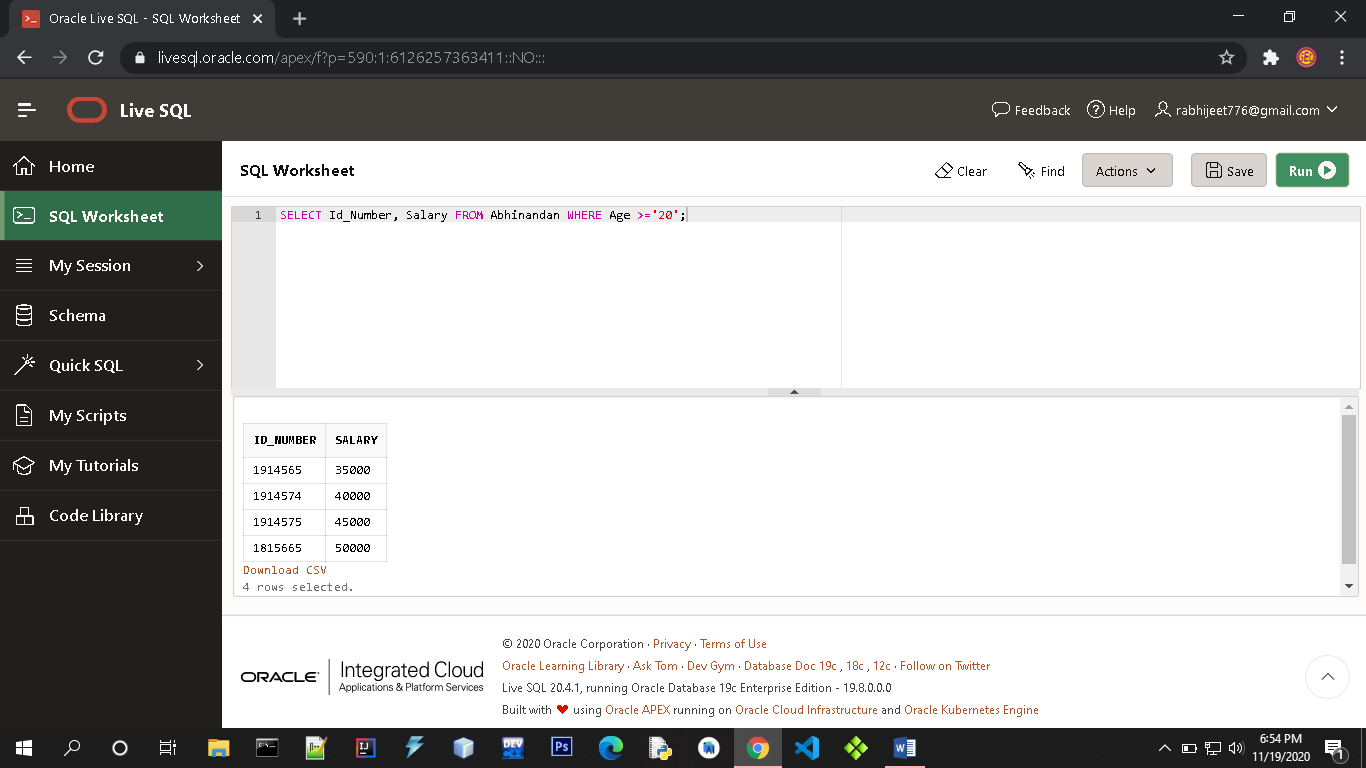
|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** |  |
| = | Equal |  |

|  |  |  |
| --- | --- | --- |
| > | Greater than |  |
| < | Less than |  |
| >= | Greater than or equal |  |
| <= | Less than or equal |  |
| <> | Not equal | | |  |
| BETWEEN | Between a certain range |  |
| LIKE | Search for a pattern |  |
| IN | To specify multiple possible values for a column |  |
|  |  |  |

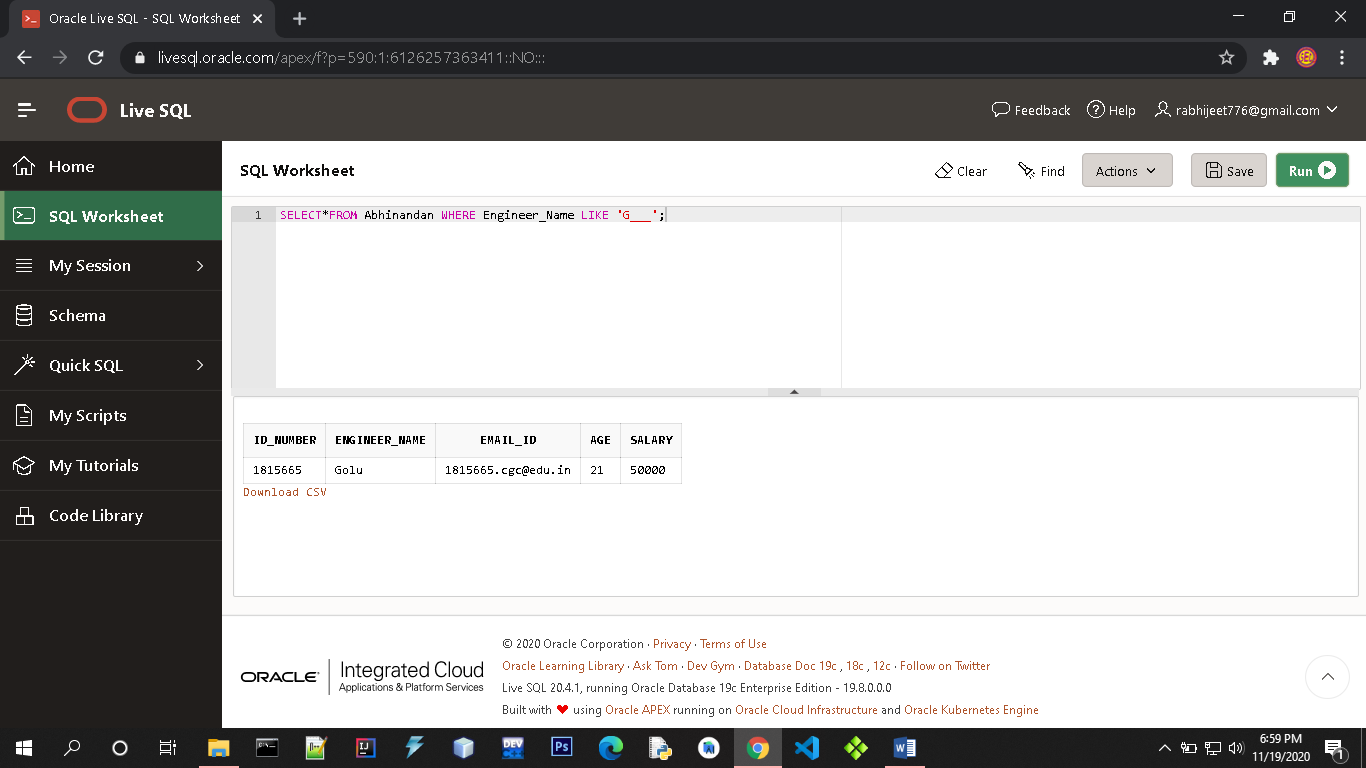
EXAMPLE of Equal(=) :-



EXAMPLE of Greater or Equal(>=) :-



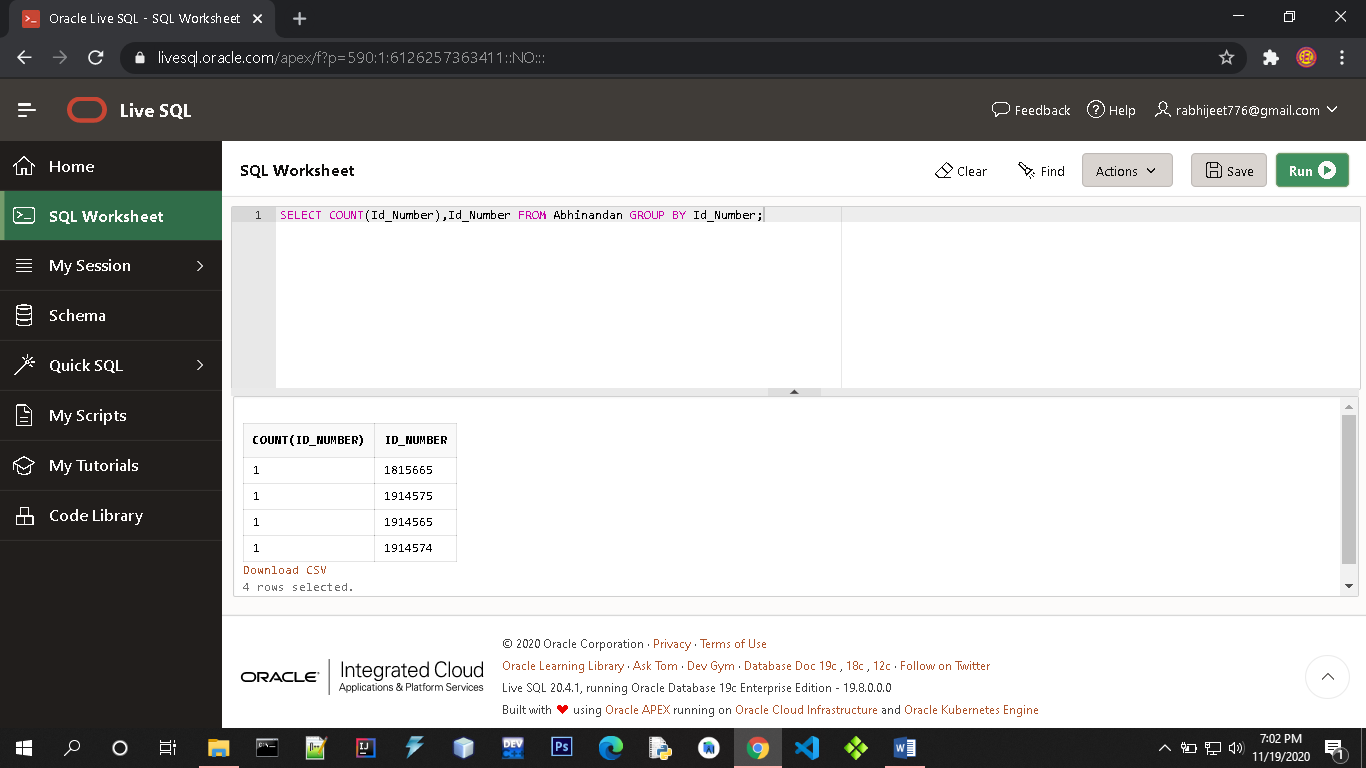
EXAMPLE of Like :-



**B) GROUP BY CLAUSE**

The GROUP BY statement groups the rows that have the same values into summary rows, like "find the number of customers in each country".

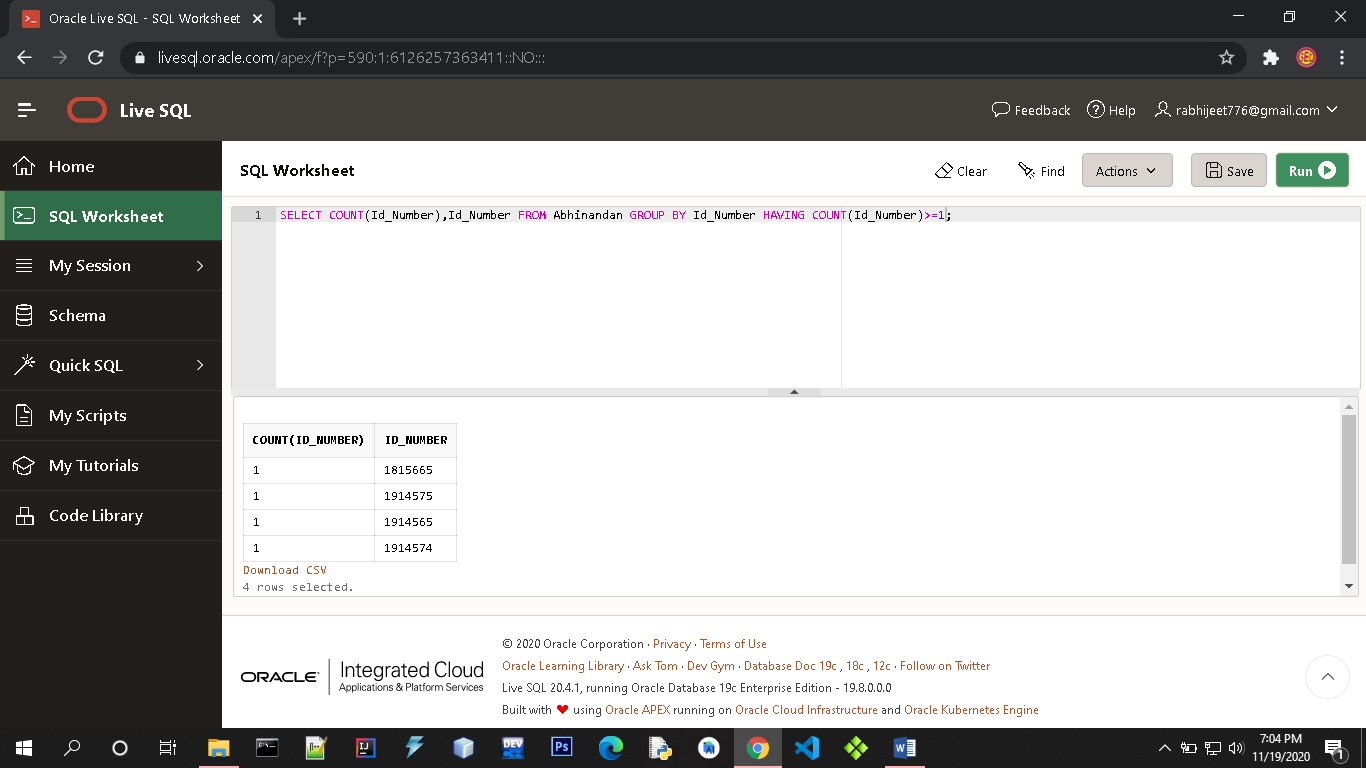
Syntax- SELECT *column\_name(s)*  
 FROM *table\_name*  
 WHERE *condition*  
 GROUP BY *column\_name(s)* ORDER BY *column\_name(s);*



**C) HAVING CLAUSE**

The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.

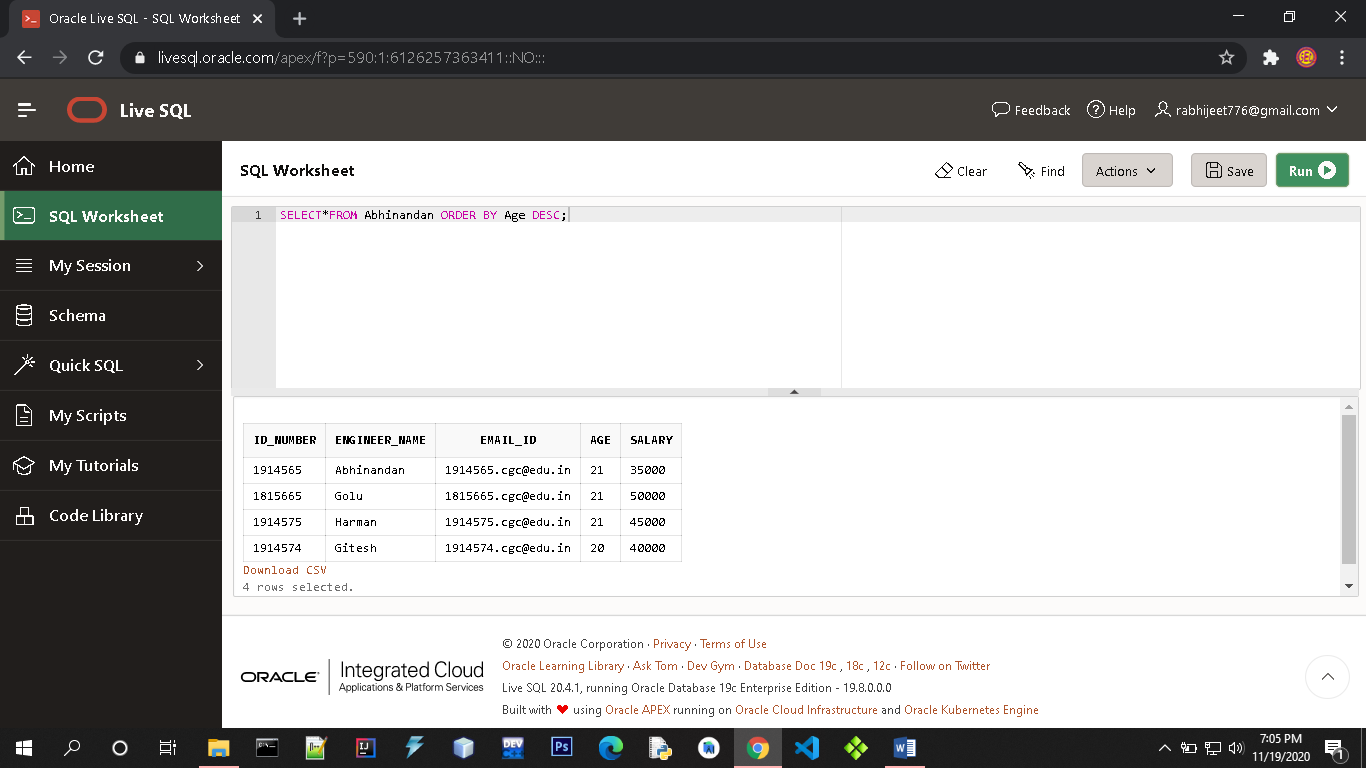
**SYNTAX:** SELECT *column\_name(s)*  
 FROM *table\_name*   
 WHERE *condition*  
 GROUP BY *column\_name(s)* HAVING *condition* ORDER BY *column\_name(s);*



**D) ORDER BY CLAUSE**

The ORDER BY keyword is used to sort the result-set in ascending or descending order.

**SYNTAX:**  SELECT *column1*, *column2, ...*  
 FROM *table\_name*  
 ORDER BY *column1, column2, ...* ASC|DESC;



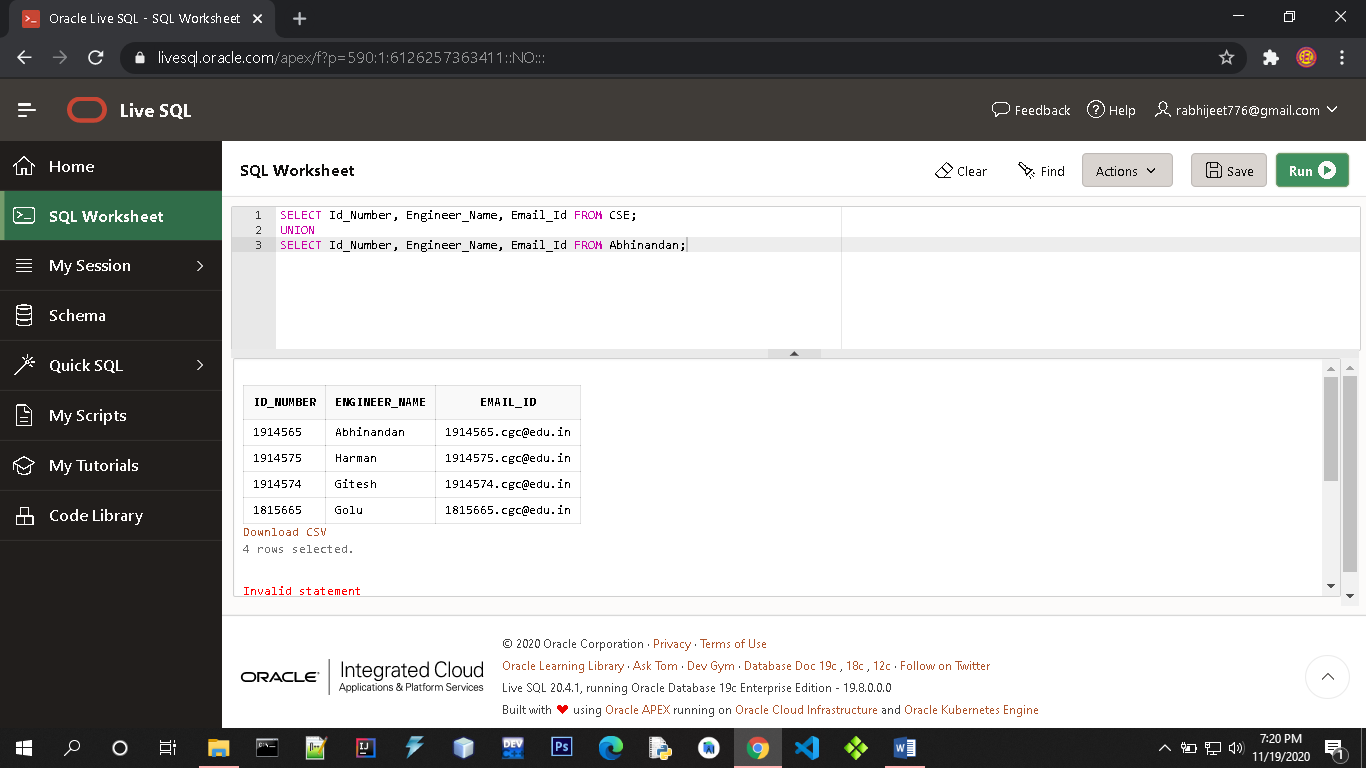
**E) UNION CLAUSE**

The UNION operator is used to combine the result-set of two or more SELECT statements.

* Each SELECT statement within UNION must have the same number of columns
* The columns must also have similar data types
* The columns in each SELECT statement must also be in the same order

**UNION Syntax:**

SELECT *column\_name(s)* FROM *table1*  
UNION  
SELECT *column\_name(s)* FROM *table2*;



**F) INTERSECTION CLAUSE**

The SQL **INTERSECT** clause/operator is used to combine two SELECT statements, but returns rows only from the first SELECT statement that are identical to a row in the second SELECT statement. This means INTERSECT returns only common rows returned by the two SELECT statements.

**SYNTAX:**  SELECT column1 [, column2 ]

FROM table1 [, table2 ]

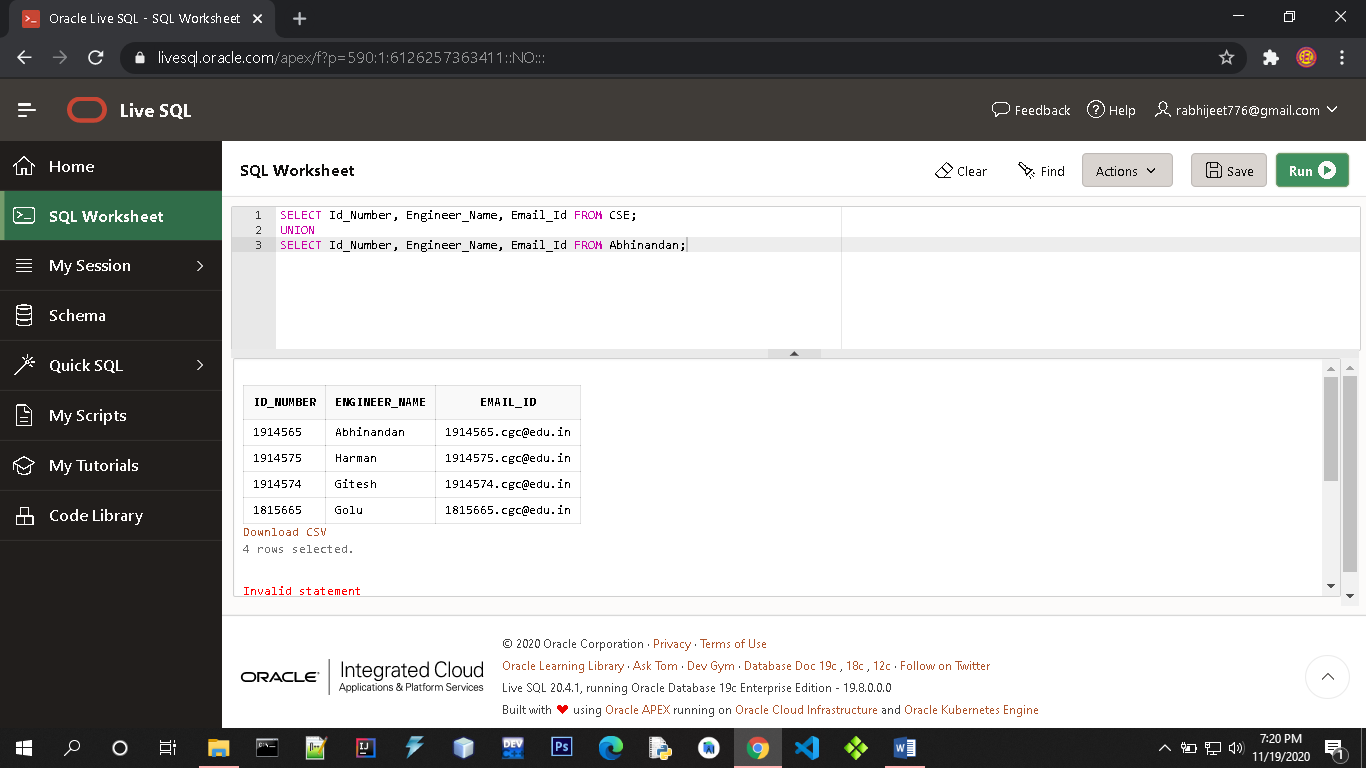
[WHERE condition]

INTERSECT

SELECT column1 [, column2 ]

FROM table1 [, table2 ]

[WHERE condition]



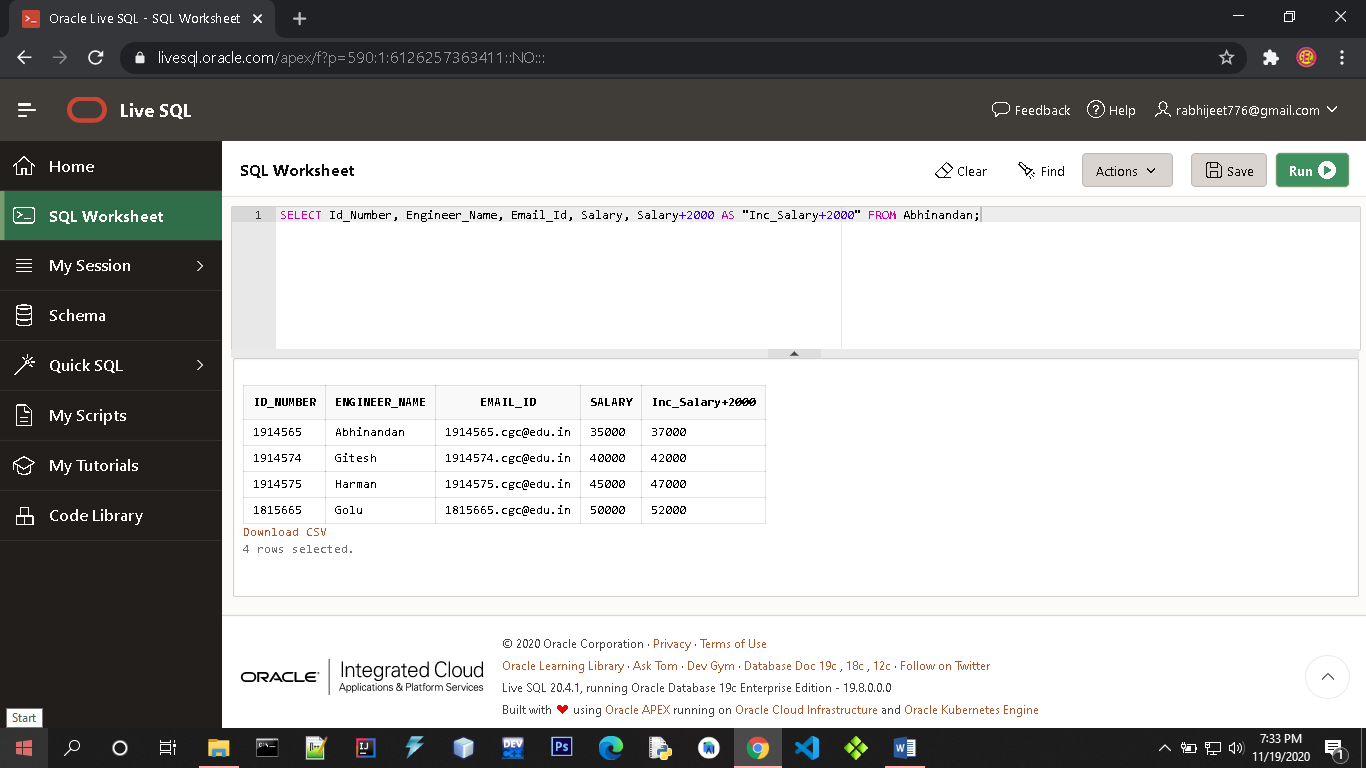
**PRACTICAL :- 06**

**Aim :- To perform various arithmetic operators and logical operators on database**

**1 .ARITHMETIC OPERATORS:**

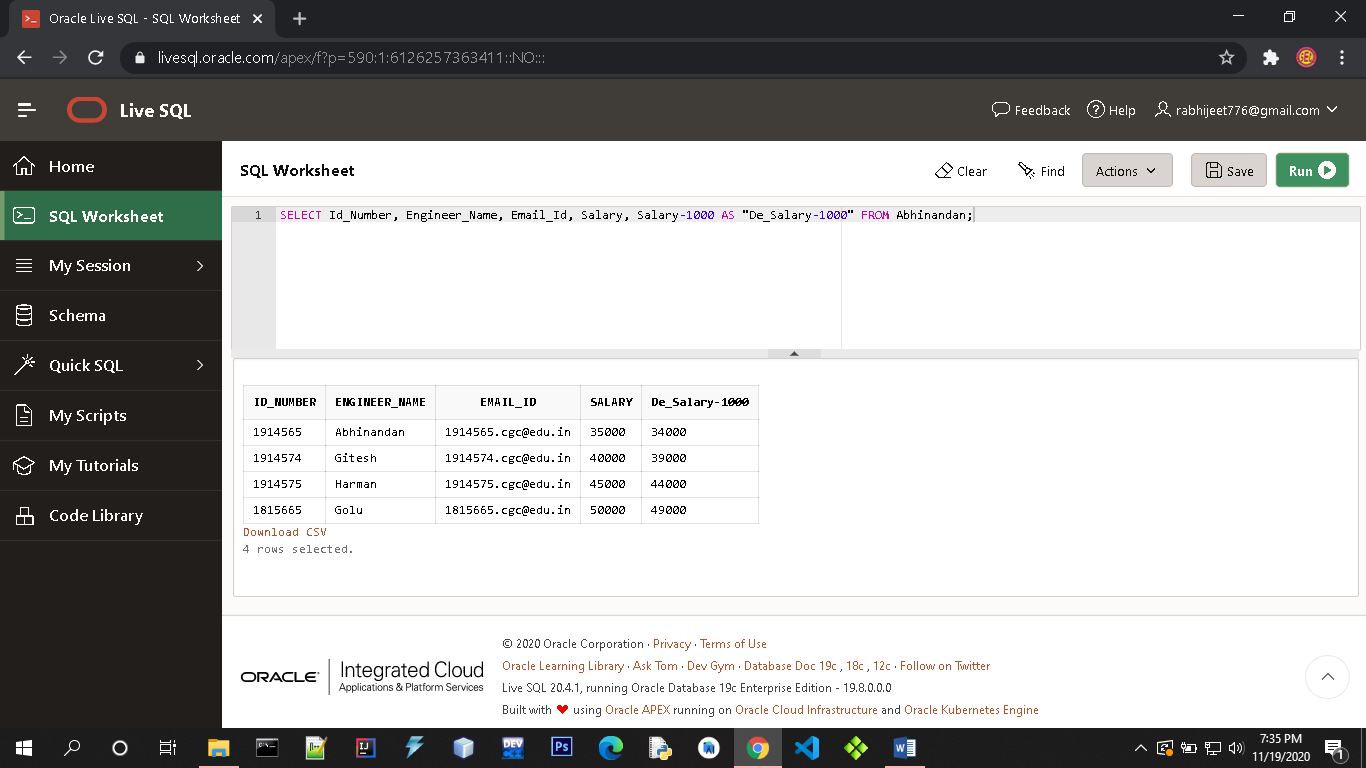
**a) Addition (+)🡪**

It is used to perform **addition operation** on data items, items include either single column or multiple columns.

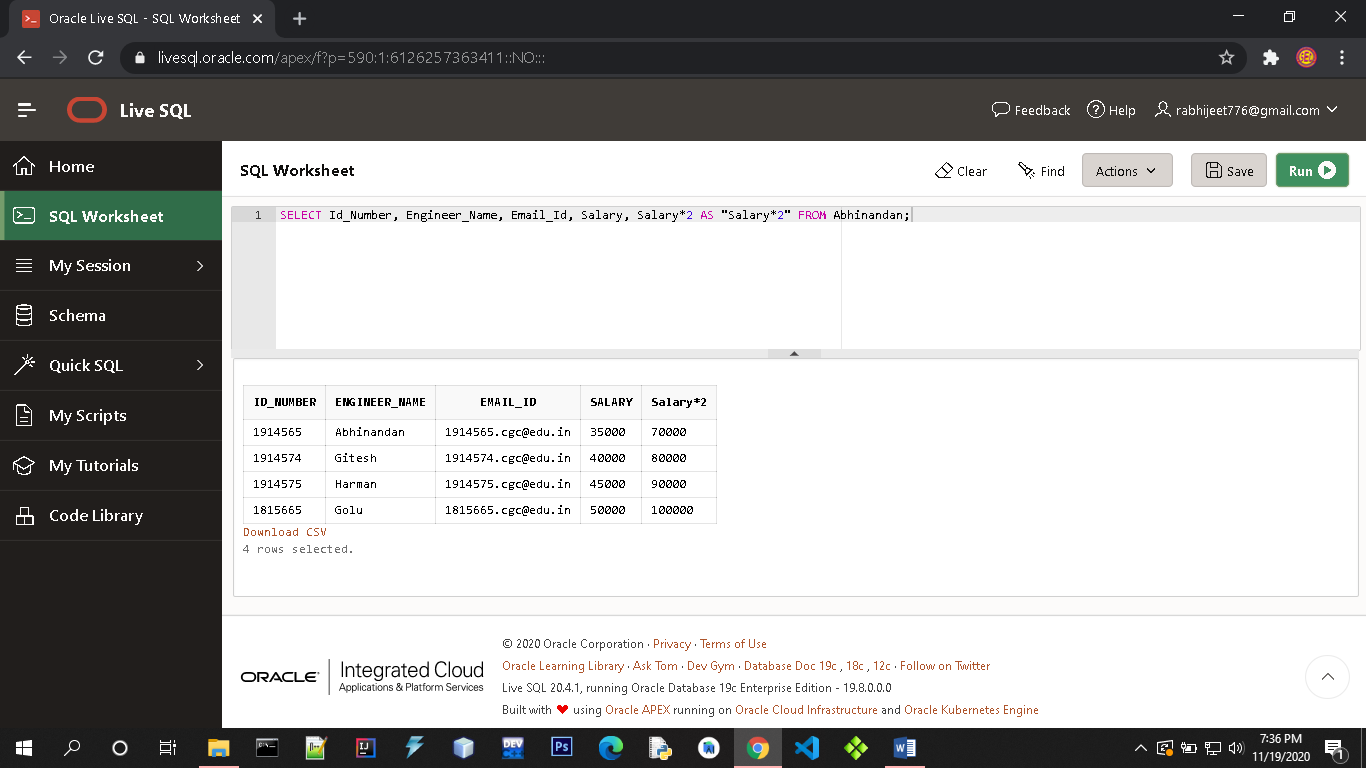


**b) Subtraction (-)🡪**

It is use to perform **subtraction operation** on the data items, items include either single column or multiple columns.

****

**c) Multiplication (\*)🡪** It is use to perform **multiplication** of data items

****

**d) Division (/)🡪** It is use to perform **division** of data items.

****

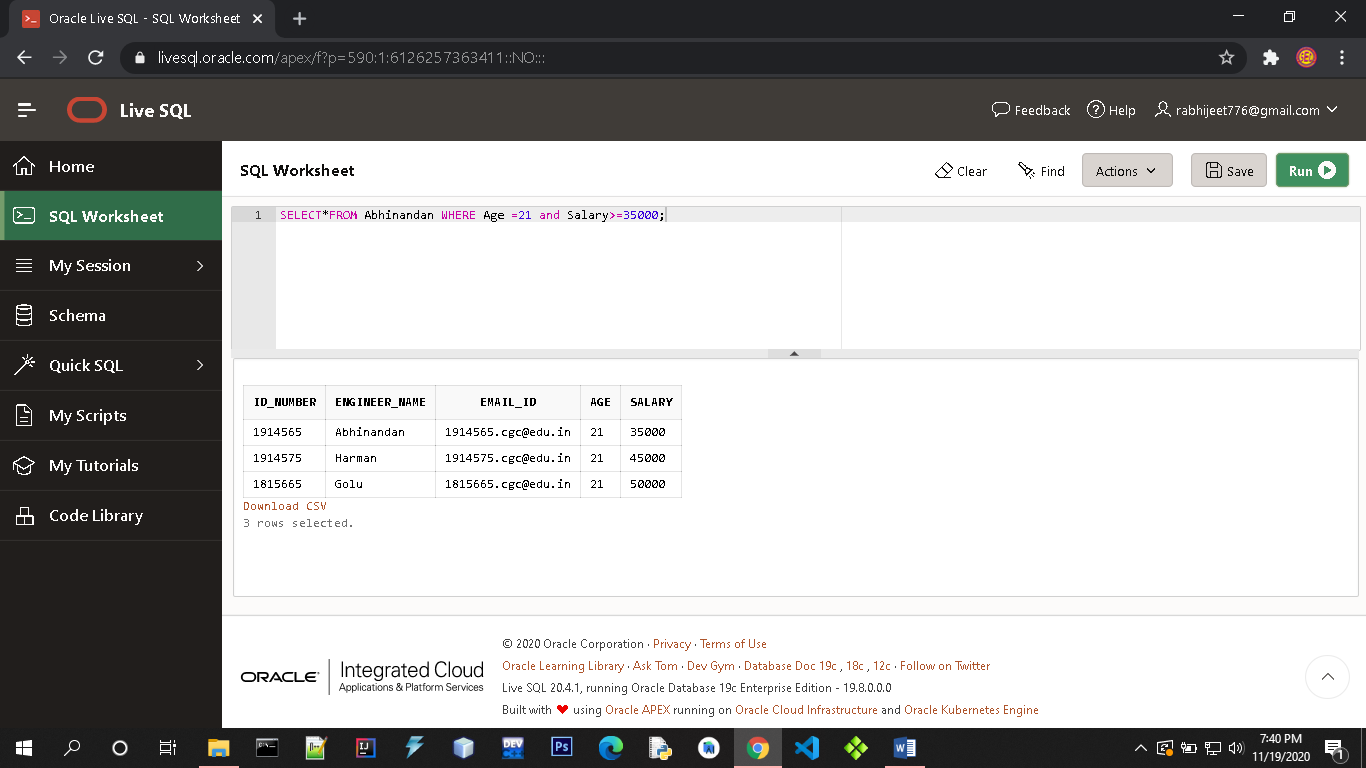
**2. LOGICAL OPERATORS:**

**a) AND OPERATOR🡪** This operator displays only those records where both the conditions **condition1 and condition2 evaluates to True.**

**SELECT \* FROM table\_name WHERE condition1 AND condition2 and ... condition N;**

**table\_name**: name of the table

**condition1,2,.. N** : first condition, second condition and so on

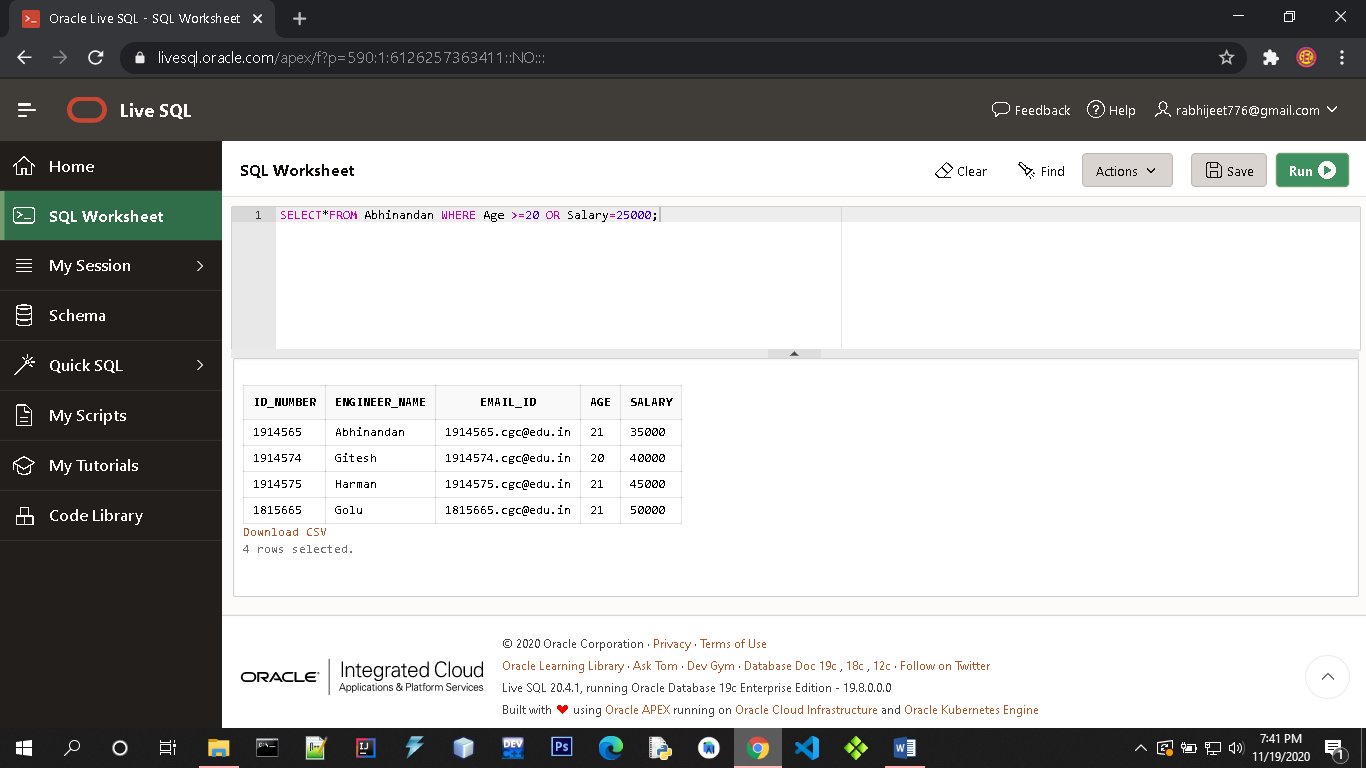


**b) OR OPERATOR🡪** This operator displays the records where either one of the conditions condition1 and condition2 evaluates to True. That is,**either condition1 is True or condition2 is True.**

**SELECT \* FROM table\_name WHERE condition1 OR condition2 OR... condition N;**

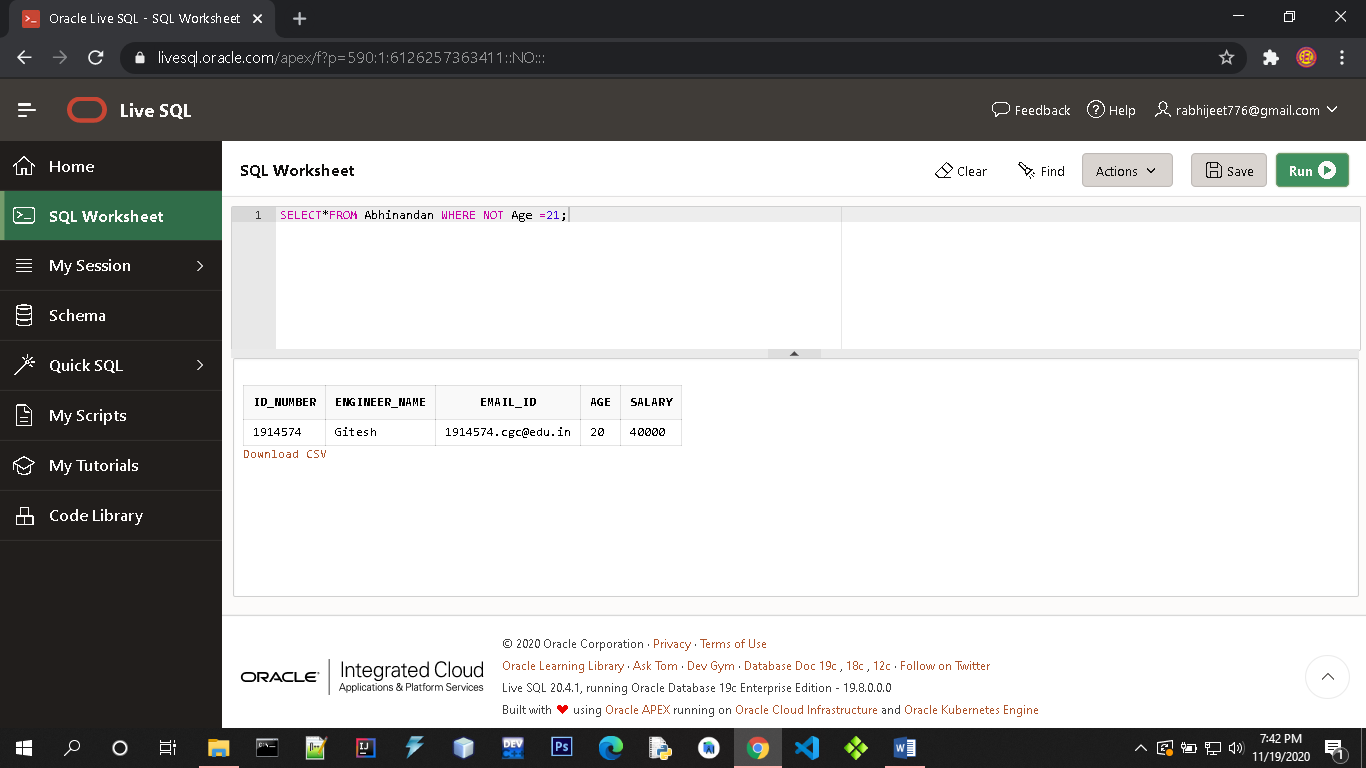
**table\_name**: name of the table

**condition1,2,..N** : first condition, second condition and so on

****

**C) NOT OPERATOR 🡪**

**NOT Syntax :** SELECT column1, colomn2, … FROM table\_name WHERE NOT condition;

****

**PRACTICAL :- 07**

**Aim :- To Study Differnet Constraints.**

**Constraints:-**

Constraints are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the table. If there is any violation between the constraint and the data action, the action is aborted.

Constraints can be column level or table level. Column level constraints apply to a column, and table level constraints apply to the whole table.

**Syntax:-**

CREATE TABLE table\_name (  
    column1 datatype *constraint*,  
    column2 datatype *constraint*,  
    column3 datatype *constraint*,  
    ....  
);

The following constraints are commonly used in SQL:

[**NOT NULL**](https://www.w3schools.com/sql/sql_notnull.asp)**:-**

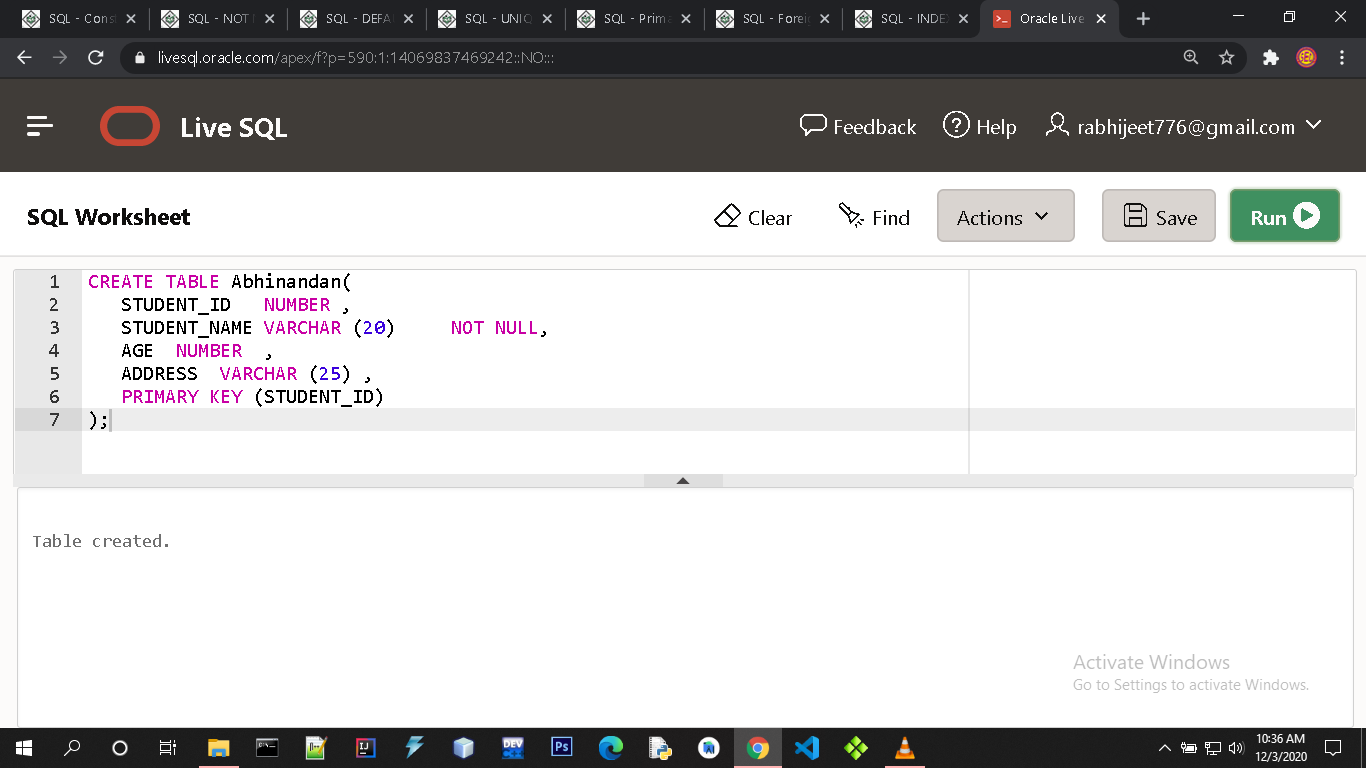
By default, a column can hold NULL values. If you do not want a column to have a NULL value, then you need to define such a constraint on this column specifying that NULL is now not allowed for that column.

**Syntax:**

CREATE TABLE tableName (

column1 datatype *NOT NULL*,  
    column2 datatype *NOT NULL*,  
    column3 datatype *NOT NULL*

);



**Primary Key:-**

A primary key is a field in a table which uniquely identifies each row/record in a database table. Primary keys must contain unique values. A primary key column cannot have NULL values.

A table can have only one primary key, which may consist of single or multiple fields. When multiple fields are used as a primary key, they are called a composite key.

If a table has a primary key defined on any field(s), then you cannot have two records having the same value of that field(s).

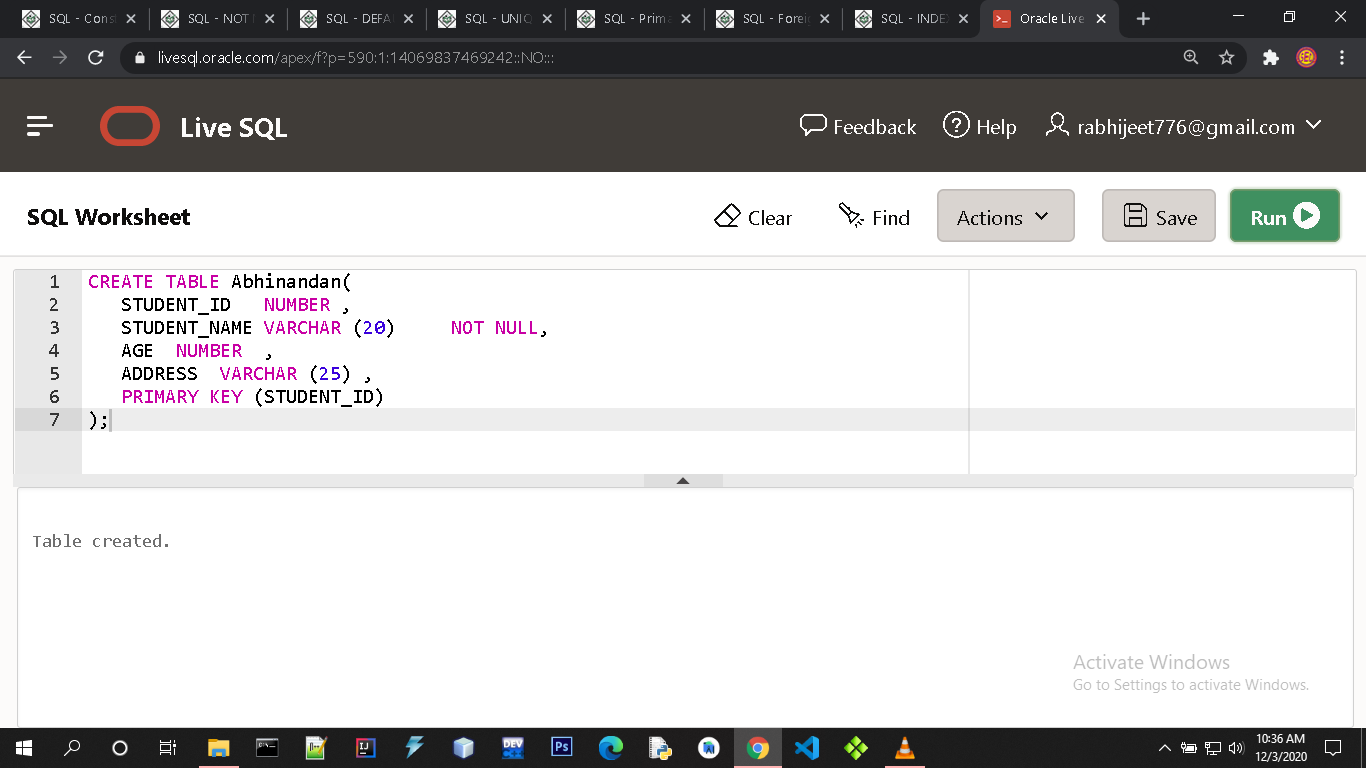
**Syntax:-**

CREATE TABLE tableName (

column1 datatype ,  
    column2 datatype *,*  
    column3 datatype ,

Primary key(column1)

);



**Foreign Key:-**

A foreign key is a key used to link two tables together. This is sometimes also called as a referencing key.

A Foreign Key is a column or a combination of columns whose values match a Primary Key in a different table.

The relationship between 2 tables matches the Primary Key in one of the tables with a Foreign Key in the second table.

If a table has a primary key defined on any field(s), then you cannot have two records having the same value of that field(s).

**Syntax:-**

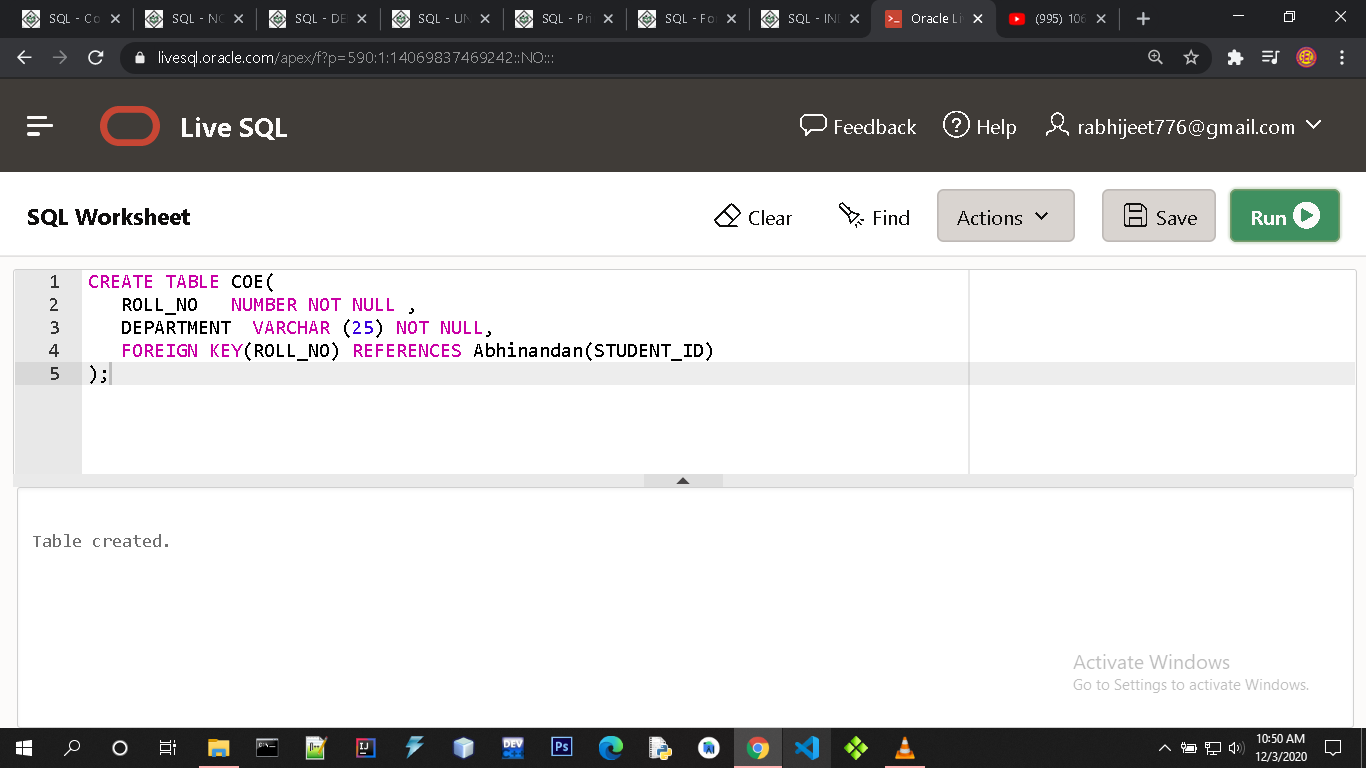
CREATE TABLE tableName (

column1 datatype ,

column2 datatype *,*  
  column3 datatype ,

FOREIGN KEY(column1)REFERENCES Another\_tablename(column)

);



**DEFAULT constraint:-**

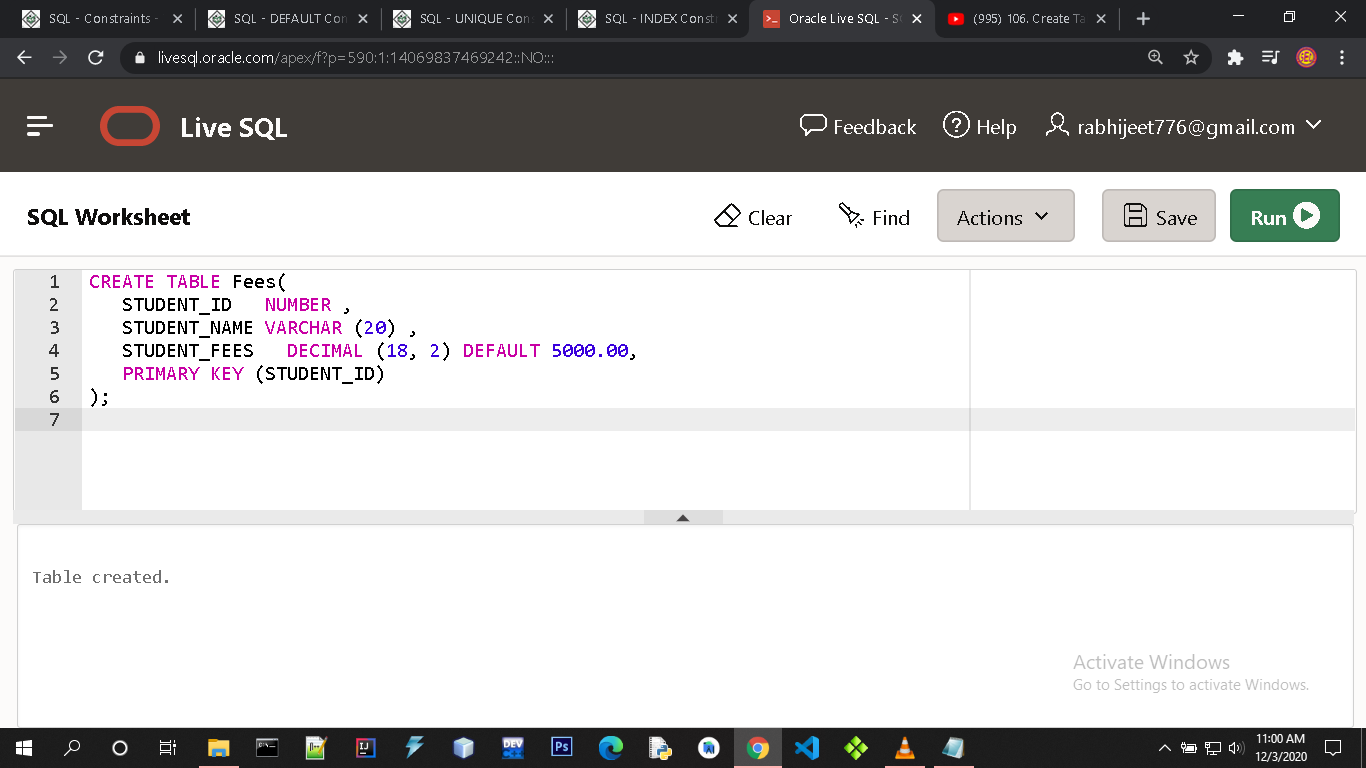
The DEFAULT constraint provides a default value to a column when the INSERT INTO statement does not provide a specific value.

**Syntax:-**

CREATE TABLE tableName (

column1 datatype ,  
column2 datatype *,*  
column3 datatype Default value

);



**UNIQUE Constraint:-**

The UNIQUE Constraint prevents two records from having identical values in a column. In the CUSTOMERS table, for example, you might want to prevent two or more people from having an identical age.

**Syntax:-**

CREATE TABLE tableName (

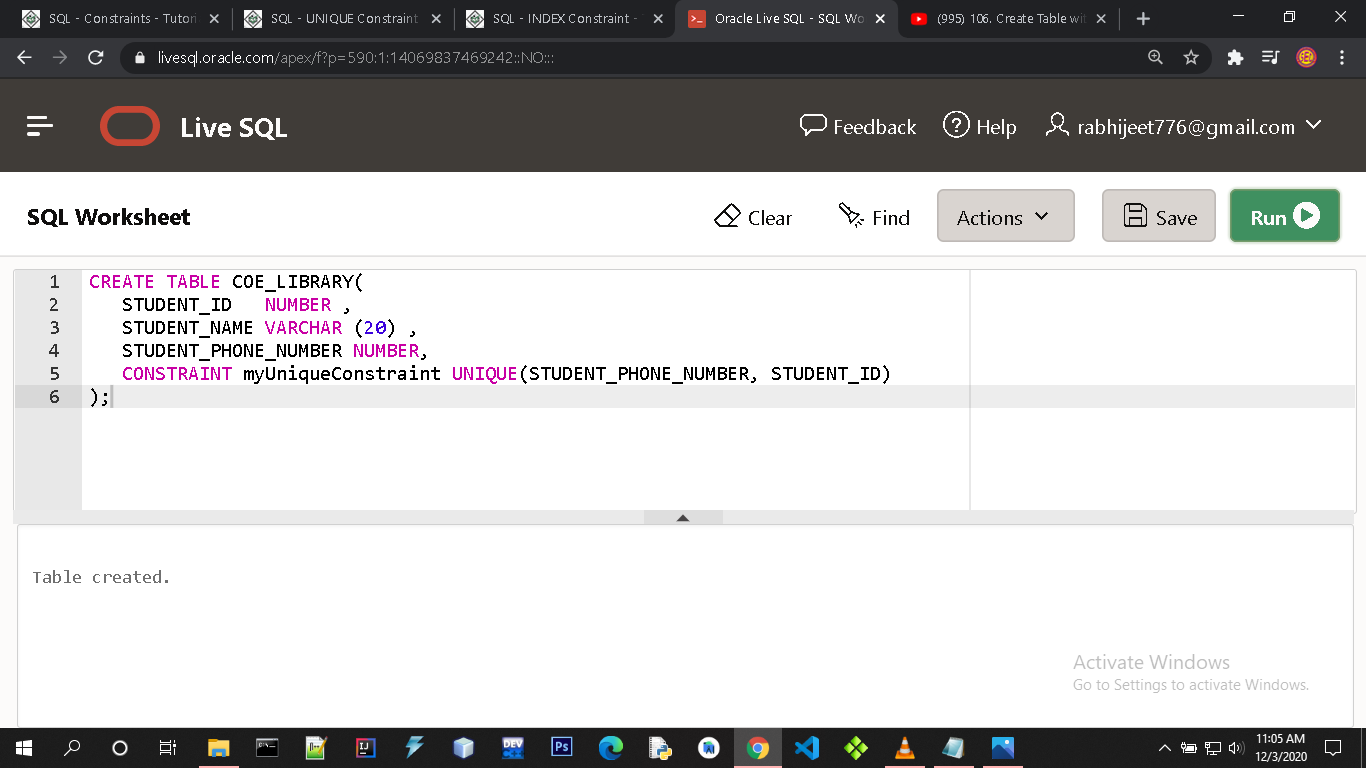
column1 datatype ,

column2 datatype *,*

column3 datatype ,

CONSTRAINT myUniqueConstraint UNIQUE(column2)

);



**INDEX :-**

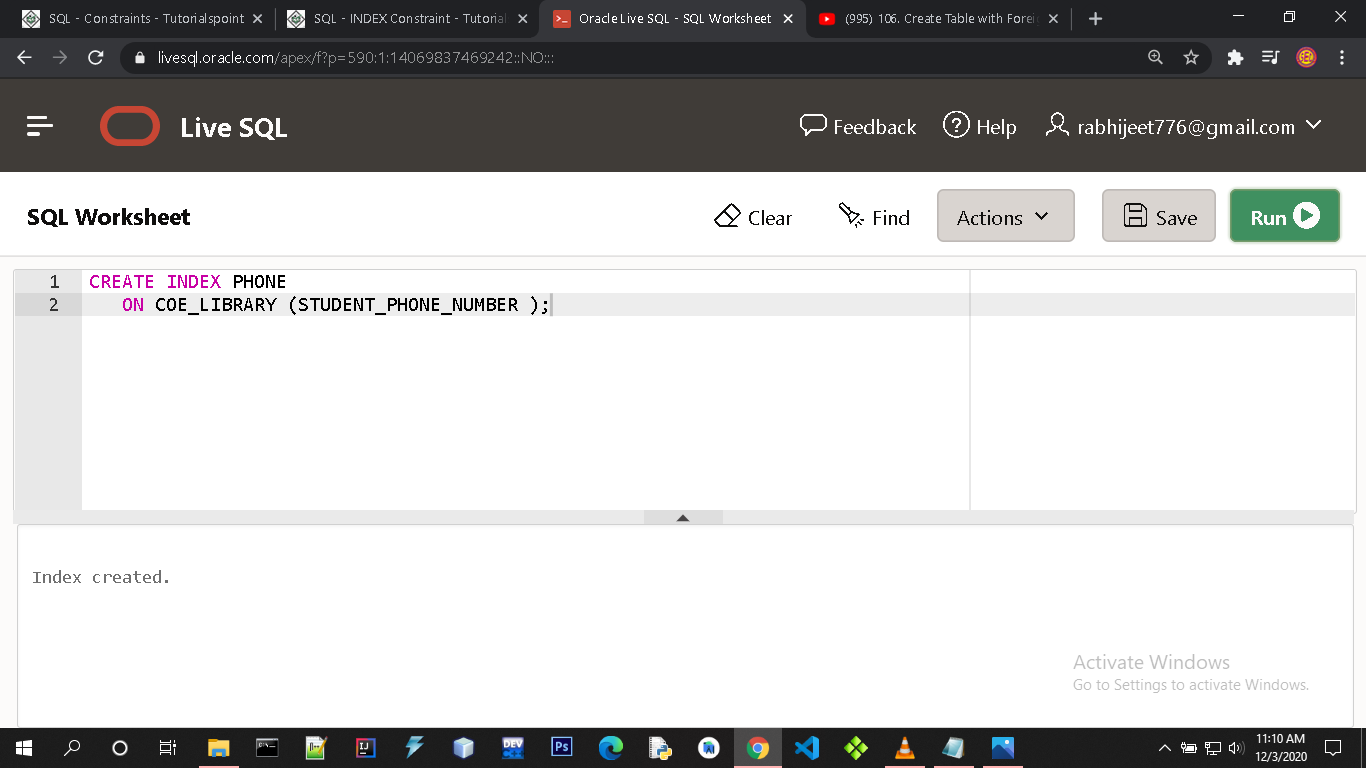
The INDEX is used to create and retrieve data from the database very quickly. An Index can be created by using a single or group of columns in a table. When the index is created, it is assigned a ROWID for each row before it sorts out the data.

Proper indexes are good for performance in large databases, but you need to be careful while creating an index. A Selection of fields depends on what you are using in your SQL queries.

**Syntax:-**

CREATE INDEX IndexName

ON TableName(columnName);



**CHECK Constraint:-**

The CHECK Constraint enables a condition to check the value being entered into a record. If the condition evaluates to false, the record violates the constraint and isn't entered the table.

**Syntax:-**

CREATE TABLE tableName (

column1 datatype ,  
column2 datatype ,  
column3 datatype CHECK (Condition)

);

