```
DROP TABLE STUDENTINFO PURGE;
CREATE TABLE STUDENTINFO(
    STUDENT_ID NUMBER(10) PRIMARY KEY,
    FIRST_NAME VARCHAR2(200),
    LAST_NAME
     VARCHAR2(200),
     DATE_OF_BIRTH DATE,
     GENDER VARCHAR2(200),
     EMAIL VARCHAR2(250) UNIQUE,
     PHONE_NUMBER VARCHAR2(10)
);

INSERT INTO STUDENTINFO
VALUES(101,'JOHN','SMITH','15-MAY-1998','M','john.smith@email.com',1234567890);
INSERT INTO STUDENTINFO
VALUES(102,'EMILY','DAVIS','20-MARCH-1999','M','emily.davis@email.com',9876543210);
INSERT INTO STUDENTINFO
VALUES(103,'MICHAEL','JOHNSON','10-JULY-1997','M','michael.johnson@email.com',5551234567);
INSERT INTO STUDENTINFO
VALUES(104,'SARAH','WILSON','05-JAN-2000','F','sarah.wilson@email.com',7899876543);
INSERT INTO STUDENTINFO
VALUES(105,'DAVID','BROWN','30-SEP-1996','M','david.brown@email.com',1112223333);
INSERT INTO STUDENTINFO
VALUES(106,'OLIVIA','LEE','18-DEC-1999','F','olivia.lee@email.com',4445556666);
INSERT INTO STUDENTINFO
VALUES(107,'ETHAN','MARTINEZ','25-NOV-1998','M','ethan.martinez@email.com',7778889999);
INSERT INTO STUDENTINFO
VALUES(108,'SOPHIA','TAYLOR','14-FEB-2002','F','sophia.taylor@email.com',2223334444);
INSERT INTO STUDENTINFO
VALUES(109,'AIDEN','MILLER','12-APR-1997','M','aiden.miller@email.com',6667778888);
```

```
INSERT INTO STUDENTINFO
VALUES(110,'EMMA','ANDERSON','07-AUG-2002','F','emma.anderson@email.com
',3334445555);
INSERT INTO STUDENTINFO
VALUES(111,'BENJAMIN','HARRIS','22-JUN-1995','M','benjamin.harris@email
.com',8889990000);
INSERT INTO STUDENTINFO
VALUES(112,'MIA','JOHNSON','01-OCT-1998','F','mia.johnson@email.com',99
90001111);
INSERT INTO STUDENTINFO
VALUES(113,'WILLIAM','WHITE','12-MARCH-2003','M','willain.white@email.c
om',0001112222);
INSERT INTO STUDENTINFO
VALUES(114,'AVA','ROBINSON','28-APR-1999','F','ava.robinson@email.com',
1112223333);
INSERT INTO STUDENTINFO
VALUES(115,'JAMES','TURNER','05-DEC-1996','M','james.turner@email.com',
2223334444);

1 row(s) inserted.
```

```
 SELECT * FROM STUDENTINFO;

 STUDENT_ID     FIRST_NAME     LAST_NAME
     DATE_OF_BIRTH     GENDER                              EMAIL
PHONE_NUMBER

 ------------- -------------- ------------- ----------------- ----------

------------------------------------------------------------
            101  JOHN            SMITH         15-05-98          M
john.smith@email.com                1234567890
            102  EMILY           DAVIS         20-03-99          M
emily.davis@email.com               9876543210
            103 MICHAEL          JOHNSON       10-07-97          M
michael.johnson@email.com           5551234567
            104  SARAH           WILSON        05-01-00          F
sarah.wilson@email.com              7899876543
            105 DAVID            BROWN         30-09-96          M
david.brown@email.com               1112223333
```

```
         106 OLIVIA         LEE         18-12-99         F
olivia.lee@email.com         4445556666
         107 ETHAN          MARTINEZ    25-11-98         M
ethan.martinez@email.com     7778889999
         108 SOPHIA         TAYLOR      14-02-02         F
sophia.taylor@email.com      2223334444
         109 AIDEN          MILLER      12-04-97         M
aiden.miller@email.com       6667778888
         110 EMMA           ANDERSON    07-08-02         F
emma.anderson@email.com      3334445555
         111 BENJAMIN       HARRIS      22-06-95         M
benjamin.harris@email.com    8889990000
         112 MIA            JOHNSON     01-10-98         F
mia.johnson@email.com        9990001111
         113 WILLIAM        WHITE       12-03-03         M
willain.white@email.com      1112222
         114 AVA            ROBINSON    28-04-99         F
ava.robinson@email.com       1112223333

   STUDENT_ID    FIRST_NAME    LAST_NAME
       DATE_OF_BIRTH     GENDER                          EMAIL
PHONE_NUMBER
__ __________ _____________ ____________ ________________ _________
_________________________ _______________
         115 JAMES          TURNER      05-12-96         M
james.turner@email.com       2223334444

15 rows selected.
```

-- --------------------------

ASCII and CHR Functions:

1. Retrieve the student_id, first_name, and LAST_NAME from the StudentInfo table. Use the ASCII function to find the ASCII values of the first characters of both first and last names for each student.

```
SELECT STUDENT_ID,ASCII(FIRST_NAME)AS FIRST_NAME,ASCII(LAST_NAME
)AS LAST_NAME FROM STUDENTINFO;
   STUDENT_ID    FIRST_NAME    LAST_NAME
```

```
------------- ------------- ------------
          101            74           83
          102            69           68
          103            77           74
          104            83           87
          105            68           66
          106            79           76
          107            69           77
          108            83           84
          109            65           77
          110            69           65
          111            66           72
          112            77           74
          113            87           87
          114            65           82

   STUDENT_ID    FIRST_NAME    LAST_NAME


-- ---------- ------------- ------------
          115            74           84

15 rows selected.
SET PAGE SIZE 50;

SELECT
  student_id,
  email,
  ASCII('@') AS ascii_at
FROM StudentInfo;

2.SELECT STUDENT_ID,EMAIL,ASCII('@')AS ASCII_EMAIL FROM STUDENTINFO;


   STUDENT_ID                         EMAIL    ASCII_EMAIL

------------- ---------------------------- --------------
          101 john.smith@email.com                     64
          102 emily.davis@email.com                    64
          103 michael.johnson@email.com                64
          104 sarah.wilson@email.com                   64
```

```
       105 david.brown@email.com                        64
       106 olivia.lee@email.com                         64
       107 ethan.martinez@email.com                     64
       108 sophia.taylor@email.com                      64
       109 aiden.miller@email.com                       64
       110 emma.anderson@email.com                      64
       111 benjamin.harris@email.com                    64
       112 mia.johnson@email.com                        64
       113 willain.white@email.com                      64
       114 ava.robinson@email.com                       64
       115 james.turner@email.com                       64

15 rows selected.


   STUDENT_ID                         EMAIL    ASCII_EMAIL
------------- ---------------------------- --------------
          101 john.smith@email.com                     64
          102 emily.davis@email.com                    64
          103 michael.johnson@email.com                64
          104 sarah.wilson@email.com                   64
          105 david.brown@email.com                    64
          106 olivia.lee@email.com                     64
          107 ethan.martinez@email.com                 64
          108 sophia.taylor@email.com                  64
          109 aiden.miller@email.com                   64
          110 emma.anderson@email.com                  64
          111 benjamin.harris@email.com                64
          112 mia.johnson@email.com                    64
          113 willain.white@email.com                  64
          114 ava.robinson@email.com                   64
          115 james.turner@email.com                   64

15 rows selected.
```

3.Display the STUDENT_ID and first_name from the StudentInfo table. Use the CHR function to create a new column containing a special character for each student, such as a heart symbol (♥).

```
SELECT STUDENT_ID, FIRST_NAME , CHR(9829) AS SPECIAL_CHARACTER FROM
STUDENTINFO;


   STUDENT_ID     FIRST_NAME    SPECIAL_CHARACTER

 _____________ _____________ ____________________
          101 JOHN          &e
          102 EMILY         &e
          103 MICHAEL       &e
          104 SARAH         &e
          105 DAVID         &e
          106 OLIVIA        &e
          107 ETHAN         &e
          108 SOPHIA        &e
          109 AIDEN         &e
          110 EMMA          &e
          111 BENJAMIN      &e
          112 MIA           &e
          113 WILLIAM       &e
          114 AVA           &e
          115 JAMES         &e

15 rows selected.
```

4.Calculate the sum of ASCII values for the characters in each students first name. Retrieve the student_id, first_name, and the calculated sum using the ASCII function and aggregation.

```
SELECT STUDENT_ID FIRST_NAME,
       SUM(ASCII(SUBSTR(FIRST_NAME, 1, 1)) +
           ASCII(SUBSTR(FIRST_NAME, 2, 1)) +
           ASCII(SUBSTR(FIRST_NAME, 3, 1)) +
           ASCII(SUBSTR(FIRST_NAME, 4, 1)) +
           ASCII(SUBSTR(FIRST_NAME, 5, 1))
           ) AS sum_of_ascii_values
FROM StudentInfo GROUP BY STUDENT_ID, FIRST_NAME;
```

```
   STUDENT_ID     FIRST_NAME     SUM_OF_ASCII_VALUES

------------- -------------
          101 JOHN
          102 EMILY                                384
          103 MICHAEL                              354
          104 SARAH                                367
          105 DAVID                                360
          106 OLIVIA                               387
          107 ETHAN                                368
          108 SOPHIA                               387
          109 AIDEN                                353
          110 EMMA
          111 BENJAMIN                             352
          112 MIA
          113 WILLIAM                              385
          114 AVA
          115 JAMES                                368

15 rows selected.

5.Retrieve the student_id, LAST_NAME
, and the ASCII value of the last character in the last name for each
student using the ASCII function.
//SELECT STUDENT_ID,LAST_NAME
,ASCII(SUBSTR(RIGHT(LAST_NAME ,1)))AS ASCII_VALUE FROM STUDENTINFO;
-------------------------------------------------------------------------------------------------------------------

--------------------------------------------------------------------------------------------

CONCAT Function:

1.Create a query that retrieves the student_id, first_name, and
LAST_NAME
 from the StudentInfo table. Use the CONCAT function to display the
full names in the format "Last Name, First Name."
SELECT STUDENT_ID, CONCAT(LAST_NAME
,FIRST_NAME)AS FULL_NAME FROM STUDENTINFO;
   STUDENT_ID          FULL_NAME

------------- -----------------
          101 SMITH JOHN
          102 DAVISEMILY
```

```
       103 JOHNSON MICHAEL
       104 WILSON SARAH
       105 BROWN DAVID
       106 LEE OLIVIA
       107 MARTINEZ ETHAN
       108 TAYLOR SOPHIA
       109 MILLER AIDEN
       110 ANDERSON MMA
       111 HARRIS BENJAMIN
       112 JOHNSON MIA
       113 WHITE WILLIAM
       114 ROBINSON AVE
       115 TURNERJAMES

15 rows selected.
```

2.You want to create email addresses for students based on their first names. Retrieve the student_id, first_name, and a new column with email addresses using the CONCAT function. Assume the email domain is '@example.com'.

```
SELECT STUDENT_ID,FIRST_NAME, CONCAT(FIRST_NAME,'@EXAMPLE.COM')AS
EMAIL_ADDRESS FROM STUDENTINFO;
```

```
   STUDENT_ID     FIRST_NAME           EMAIL_ADDRESS

------------- ------------- ----------------------
          101 JOHN          JOHN@example.com
          102 EMILY         EMILY@example.com
          103 MICHAEL       MICHAEL@example.com
          104 SARAH         SARAH@example.com
          105 DAVID         DAVID@example.com
          106 OLIVIA        OLIVIA@example.com
          107 ETHAN         ETHAN@example.com
          108 SOPHIA        SOPHIA@example.com
          109 AIDEN         AIDEN@example.com
          110 EMMA          EMMA@example.com
          111 BENJAMIN      BENJAMIN@example.com
          112 MIA           MIA@example.com
          113 WILLIAM       WILLIAM@example.com
          114 AVA           AVA@example.com
```

```
       115 JAMES          JAMES@example.com

15 rows selected.
```

3.Display the student_id, email, and a new email address for each student created by concatenating their student_id with '@university.com' using the CONCAT function.

```
SELECT  STUDENT_ID, EMAIL, CONCAT(STUDENT_ID, '@university.com') AS
new_email FROM STUDENT_ID;

 STUDENT_ID                        EMAIL              NEW_EMAIL

_____________ ____________________________ _____________________
          101 john.smith@email.com         101@university.com
          102 emily.davis@email.com        102@university.com
          103 michael.johnson@email.com    103@university.com
          104 sarah.wilson@email.com       104@university.com
          105 david.brown@email.com        105@university.com
          106 olivia.lee@email.com         106@university.com
          107 ethan.martinez@email.com     107@university.com
          108 sophia.taylor@email.com      108@university.com
          109 aiden.miller@email.com       109@university.com
          110 emma.anderson@email.com      110@university.com
          111 benjamin.harris@email.com    111@university.com
          112 mia.johnson@email.com        112@university.com
          113 willain.white@email.com      113@university.com
          114 ava.robinson@email.com       114@university.com
          115 james.turner@email.com       115@university.com

15 rows selected.
```

4.Retrieve the student_id, first_name, and LAST_NAME from the StudentInfo table. Use the CONCAT function to create a new column displaying the first name followed by the last name without a space.

```
SELECT STUDENT_ID, FIRST_NAME, LAST_NAME
, CONCAT(FIRST_NAME, LAST_NAME ) AS full_name FROM STUDENT_ID;
   STUDENT_ID     FIRST_NAME     LAST_NAME
            FULL_NAME

__ __________ _____________ ____________ _________________
```

```
       101 JOHN          SMITH        JOHNSMITH
       102 EMILY         DAVIS        EMILYDAVIS
       103 MICHAEL       JOHNSON      MICHAEL JOHNSON
       104 SARAH         WILSON       SARAH WILSON
       105 DAVID         BROWN        DAVID BROWN
       106 OLIVIA        LEE          OLIVIA LEE
       107 ETHAN         MARTINEZ     ETHAN MARTINEZ
       108 SOPHIA        TAYLOR       SOPHIATAYLOR
       109 AIDEN         MILLER       AIDENMILLER
       110 EMMA          ANDERSON     EMMAANDERSON
       111 BENJAMIN      HARRIS       BENJAMIN HARRIS
       112 MIA           JOHNSON      MIAJOHNSON
       113 WILLIAM       WHITE        WILLIAM WHITE
       114 AVA           ROBINSON     AVAROBINSON
       115 JAMES         TURNER       JAMES TURNER

15 rows selected.
```

5. You need to generate usernames for students by combining their first names and the last two digits of their student_id. Retrieve the student_id, first_name, and the generated usernames using the CONCAT function.

```
SELECT STUDENT_ID, FIRST_NAME, CONCAT(FIRST_NAME, SUBSTR(STUDENT_ID,
1)) AS username FROM StudentInfo;
```

```
   STUDENT_ID    FIRST_NAME      USERNAME

_____________  _____________  _____________
          101 JOHN            JOHN01
          102 EMILY           EMILY02
          103 MICHAEL         MICHAEL03
          104 SARAH           SARAH04
          105 DAVID           DAVID05
          106 OLIVIA          OLIVIA06
          107 ETHAN           ETHAN07
          108 SOPHIA          SOPHIA08
          109 AIDEN           AIDEN09
          110 EMMA            EMMA10
          111 BENJAMIN        BENJAMIN11
          112 MIA            MIA12
          113 WILLIAM        WILLIAM13
          114 AVA            AVA14
          115 JAMES          JAMES15

15 rows selected.
```

LOWER and UPPER Functions:

1.Display the student_id and email from the StudentInfo table. Convert the email addresses to lowercase using the LOWER function.

```
SELECT STUDENT_ID, LOWER(EMAIL) AS lower_email FROM STUDENTINFO;


   STUDENT_ID                  LOWER_EMAIL

_____________ ____________________________
          101 john.smith@email.com
          102 emily.davis@email.com
          103 michael.johnson@email.com
          104 sarah.wilson@email.com
          105 david.brown@email.com
          106 olivia.lee@email.com
          107 ethan.martinez@email.com
          108 sophia.taylor@email.com
          109 aiden.miller@email.com
          110 emma.anderson@email.com
          111 benjamin.harris@email.com
          112 mia.johnson@email.com
          113 willain.white@email.com
          114 ava.robinson@email.com
          115 james.turner@email.com

15 rows selected.
```

2.Retrieve the student_id, first_name, and LAST_NAME from the StudentInfo table. Use the UPPER function to display the full names in uppercase.

```
SELECT STUDENT_ID, UPPER(CONCAT(FIRST_NAME, LAST_NAME
)) AS full_name_in_uppercase FROM StudentInfo;

   STUDENT_ID    FULL_NAME_IN_UPPERCASE
------------- -------------------------
          101 JOHNSMITH
          102 EMILY DAVIS
          103 MICHAEL JOHNSON
          104 SARAH WILSON
          105 DAVID BROWN
          106 OLIVIA LEE
          107 ETHAN MARTINEZ
          108 SOPHIA TAYLOR
          109 AIDEN MILLER
          110 EMMA ANDERSON
          111 BENJAMIN HARRIS
          112 MIA JOHNSON
          113 WILLIAM WHITE
          114 AVA ROBINSON
          115 JAMES TURNER

15 rows selected.
```

3.Calculate the total number of students with lowercase email addresses in the StudentInfo table using the LOWER function and COUNT aggregation.

```
SELECT COUNT(*) AS total_students_lowercase_emails FROM STUDENTINFO
WHERE EMAIL = LOWER(EMAIL);

   TOTAL_STUDENTS_LOWERCASE_EMAILS
---------------------------------------
                                     15
```

4.Retrieve the student_id, email, and first_name. Convert the email addresses to uppercase and display them alongside the original first names using the UPPER function.

```
SELECT STUDENT_ID, UPPER(EMAIL) AS upper_email, FIRST_NAME FROM
STUDENTINFO;
```

```
   STUDENT_ID                     UPPER_EMAIL     FIRST_NAME

_____________ ____________________________ _____________
          101 JOHN.SMITH@EMAIL.COM             JOHN
          102 EMILY.DAVIS@EMAIL.COM            EMILY
          103 MICHAEL.JOHNSON@EMAIL.COM        MICHAEL
          104 SARAH.WILSON@EMAIL.COM           SARAH
          105 DAVID.BROWN@EMAIL.COM            DAVID
          106 OLIVIA.LEE@EMAIL.COM             OLIVIA
          107 ETHAN.MARTINEZ@EMAIL.COM         ETHAN
          108 SOPHIA.TAYLOR@EMAIL.COM          SOPHIA
          109 AIDEN.MILLER@EMAIL.COM           AIDEN
          110 EMMA.ANDERSON@EMAIL.COM          EMMA
          111 BENJAMIN.HARRIS@EMAIL.COM        BENJAMIN
          112 MIA.JOHNSON@EMAIL.COM            MIA
          113 WILLAIN.WHITE@EMAIL.COM          WILLIAM
          114 AVA.ROBINSON@EMAIL.COM           AVA
          115 JAMES.TURNER@EMAIL.COM           JAMES

15 rows selected.
```

5. You want to display the student_id, email, and LAST_NAME from the StudentInfo table. Convert the email addresses to uppercase and remove any leading and trailing spaces using the UPPER function and TRIM function.

```
SELECT STUDENT_ID, UPPER(TRIM(EMAIL)) AS upper_email, LAST_NAME FROM
STUDENTINFO;
```

```
  STUDENT_ID                     UPPER_EMAIL     LAST_NAME


________________________________________ ____________
          101 JOHN.SMITH@EMAIL.COM             SMITH
          102 EMILY.DAVIS@EMAIL.COM            DAVIS
          103 MICHAEL.JOHNSON@EMAIL.COM        JOHNSON
          104 SARAH.WILSON@EMAIL.COM           WILSON
          105 DAVID.BROWN@EMAIL.COM            BROWN
          106 OLIVIA.LEE@EMAIL.COM             LEE
          107 ETHAN.MARTINEZ@EMAIL.COM         MARTINEZ
          108 SOPHIA.TAYLOR@EMAIL.COM          TAYLOR
```

```
   109 AIDEN.MILLER@EMAIL.COM        MILLER
   110 EMMA.ANDERSON@EMAIL.COM       ANDERSON
   111 BENJAMIN.HARRIS@EMAIL.COM     HARRIS
   112 MIA.JOHNSON@EMAIL.COM         JOHNSON
   113 WILLAIN.WHITE@EMAIL.COM       WHITE
   114 AVA.ROBINSON@EMAIL.COM        ROBINSON
   115 JAMES.TURNER@EMAIL.COM        TURNER

15 rows selected.
```

COUNT, AVG, MAX, MEDIAN, MIN, and SUM Functions:-

1. Calculate the total count of students in the STUDENTINFO table.

```
SELECT COUNT(*) AS total_students FROM STUDENTINFO;

   TOTAL_STUDENTS
-----------------
               15
```

2. Determine the average age of students based on their date of birth and display it.

```
SELECT AVG(TRUNC(MONTHS_BETWEEN(SYSDATE, date_of_birth) / 12)) AS
average_age FROM STUDENTINFO;

   AVERAGE_AGE
--------------
          24.2
```

3. Find the maximum and minimum lengths of students email addresses and display these values.

```
SELECT MAX(LENGTH(EMAIL)) AS max_email_length, MIN(LENGTH(EMAIL)) AS
min_email_length FROM STUDENTINFO;


   MAX_EMAIL_LENGTH    MIN_EMAIL_LENGTH
```

```
------------------- -------------------
                 25                  20
```

4. Determine the sum of ASCII values of the first character of each student s last name and display the result.

```
SELECT SUM(ASCII(SUBSTR(LAST_NAME, 1, 1))) AS sum_of_ascii_values FROM STUDENTINFO;
```

```
   SUM_OF_ASCII_VALUES
----------------------
                  1156
```

TRIM Function:

1. You have a column named DESCRIPTION in a table that contains text data. You want to remove any leading and trailing spaces from the values in this column. Write an SQL query using the TRIM function to achieve this.

```
UPDATE STUDENTINFO SET DESCRIPTION = TRIM(BOTH FROM DESCRIPTION);

SELECT TRIM('DESCRIPTION' ) AS PRODUCT_NAME FROM STUDENTINFO;
```

```
   PRODUCT_NAME
---------------
DESCRIPTION
DESCRIPTION
DESCRIPTION
DESCRIPTION
DESCRIPTION
DESCRIPTION
DESCRIPTION
DESCRIPTION
DESCRIPTION
DESCRIPTION
DESCRIPTION
```

```
DESCRIPTION
DESCRIPTION
DESCRIPTION

   PRODUCT_NAME
________________
DESCRIPTION

15 rows selected.
```

2.Retrieve the names of all students in the STUDENTINFO table. Some names have extra spaces at the beginning and end. Write an SQL query using the TRIM function to display the names without leading and trailing spaces.

```
SELECT TRIM() AS NAME FROM STUDENTINFO;
```

3.In a table that phone_number, you notice that some phone_number have unnecessary spaces. Write an SQL query using the TRIM function to remove all leading and trailing spaces from the phone_number.

```
UPDATE STUDENTINFO SET PHONE_NUMBER = TRIM(BOTH ' ' FROM PHONE_NUMBER);

15 rows updated.
```

4.You need to list all gender from a table of STUDENTINFO. However, gender have leading spaces. Write an SQL query using the TRIM function to display the gender without any leading spaces.

```
SELECT TRIM(LEADING ' ' FROM GENDER) AS GENDER FROM STUDENTINFO;


   GENDER
_________________
M
M
M
F
M
```

```
F
M
F
M
F
M
F
M
F

   GENDER
________________
M

15 rows  selected.
```

5. Retrieve a list of email addresses from the STUDENTINFO table. Some email addresses have extra spaces in them. Use the TRIM function to remove any leading and trailing spaces from the email addresses in your query.

```
SELECT TRIM(BOTH ' ' FROM EMAIL) AS EMAIL FROM STUDENTINFO;
```

```
                       EMAIL
____________________________
john.smith@email.com
emily.davis@email.com
michael.johnson@email.com
sarah.wilson@email.com
david.brown@email.com
olivia.lee@email.com
ethan.martinez@email.com
sophia.taylor@email.com
aiden.miller@email.com
emma.anderson@email.com
benjamin.harris@email.com
mia.johnson@email.com
willain.white@email.com
ava.robinson@email.com
```

```
                          EMAIL
_________________________
james.turner@email.com

15 rows selected.
```

LTRIM Function:

1. You have a column called STUDENT_ID in a table where some values have extra spaces at the beginning. Write an SQL query using the LTRIM function to remove leading spaces from the STUDENT_ID.

```
SELECT LTRIM(STUDENT_ID) AS StudentID FROM STUDENTINFO;


   STUDENTID
____________
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

15 rows selected.
```

2. In a table containing date_of_birth, some date_of_birth have leading spaces. Write an SQL query using the LTRIM function to display the date_of_birth without any leading spaces.

```
SELECT LTRIM(DATE_OF_BIRTH) AS DateOfBirth FROM STUDENTINFO;
```

```
   DATE OF BIRTH
______________
15-05-98
20-03-99
10-07-97
05-01-00
30-09-96
18-12-99
25-11-98
14-02-02
12-04-97
07-08-02
22-06-95
01-10-98
12-03-03
28-04-99
05-12-96

15 rows selected.
```

3. Retrieve a list of first_name from a table. Some first_name have leading spaces. Use the LTRIM function to remove these leading spaces in your query.

```
SELECT LTRIM(FIRST_NAME) AS FIRST_NAME FROM STUDENTINFO;
```

```
   FIRSTNAME
____________
JOHN
EMILY
MICHAEL
SARAH
```

```
DAVID
OLIVIA
ETHAN
SOPHIA
AIDEN
EMMA
BENJAMIN
MIA
WILLIAM
AVA
JAMES

15 rows selected.
```

4.You are working with data from a STUDENTINFO, and the STUDENT_IDs sometimes have extra spaces at the beginning. Write an SQL query using the LTRIM function to remove any leading spaces from the student_ IDs.

```
SELECT LTRIM(STUDENT_ID) AS StudentID FROM STUDENTINFO;


   STUDENTID
____________
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

15 rows selected.
```

5. In a table that LAST_NAME some names have leading spaces that need to be removed. Write an SQL query using the LTRIM function to clean the LAST_NAME .

```
SELECT LTRIM(LAST_NAME) AS LastName FROM STUDENTINFO;
```

```
   LAST NAME
___________
SMITH
DAVIS
JOHNSON
WILSON
BROWN
LEE
MARTINEZ
TAYLOR
MILLER
ANDERSON
HARRIS
JOHNSON
WHITE
ROBINSON
TURNER

15 rows selected.
```

```
-----------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
```

RTRIM Function:

1. You are dealing with a table that contains email_id, and some of them have trailing spaces. Write an SQL query using the RTRIM function to remove any trailing spaces from the email_id.

```
SELECT RTRIM(EMAIL) AS Email FROM STUDENTINFO;
```

```
                       EMAIL
____________________________
```

```
john.smith@email.com
emily.davis@email.com
michael.johnson@email.com
sarah.wilson@email.com
david.brown@email.com
olivia.lee@email.com
ethan.martinez@email.com
sophia.taylor@email.com
aiden.miller@email.com
emma.anderson@email.com
benjamin.harris@email.com
mia.johnson@email.com
willain.white@email.com
ava.robinson@email.com
james.turner@email.com

15 rows selected.
```

2. In a table that date_of_birth, some date have trailing spaces that need to be eliminated. Write an SQL query using the RTRIM function to display the date_of_birth without trailing spaces.

```sql
SELECT RTRIM(DATE_OF_BIRTH) AS DateOfBirth FROM STUDENTINFO;
```

```
   DATE OF BIRTH
______________
15-05-98
20-03-99
10-07-97
05-01-00
30-09-96
18-12-99
25-11-98
14-02-02
12-04-97
07-08-02
22-06-95
01-10-98
12-03-03
```

```
28-04-99
05-12-96

15 rows selected.
```

3.Retrieve a list of first_name from a table. Some first names have trailing spaces. Use the RTRIM function to remove these trailing spaces in your query.

```
SELECT RTRIM(FIRSTNAME) AS  FIRSTNAME FROM STUDENTINFO;

  FIRSTNAME
____________
JOHN
EMILY
MICHAEL
SARAH
DAVID
OLIVIA
ETHAN
SOPHIA
AIDEN
EMMA
BENJAMIN
MIA
WILLIAM
AVA
JAMES

15 rows selected.
```

5.You have a table with Students names, and some names have trailing spaces. Write an SQL query using the RTRIM function to display the student s names without any trailing spaces.

```
SELECT RTRIM(FIRST_NAME,LAST_NAME) AS STUDENTNAME FROM STUDENTINFO;

   STUDENTNAME
______________
JOHN
```

```
EMILY
MICHAEL
SARAH
DAVID
OLIVIA
ETH
SOPHI
AIDEN
EMM
BENJAMIN
MIA
WILLIAM
AVA
JAMES

15 rows selected.
```

6. In a table containing phone number, some number have trailing spaces that need to be cleaned. Write an SQL query using the RTRIM function to remove the trailing spaces from the phone number.

```
SELECT RTRIM(PHONE_NUMBER) AS PhoneNumber FROM STUDENTINFO;

   PHONE NUMBER
______________
1234567890
9876543210
5551234567
7899876543
1112223333
4445556666
7778889999
2223334444
6667778888
3334445555
8889990000
9990001111
1112222
1112223333
2223334444
```

```
15 rows selected.
```

TRIM Function:-

1. You are dealing with a table that contains Student's first name. Some first_names have both leading and trailing spaces. Write an SQL query using the TRIM function to retrieve the STUDENT_ID and cleaned first_names for all students.

```
SELECT STUDENT_ID, TRIM(BOTH ' ' FROM FIRST_NAME) AS FirstName FROM
STUDENTINFO;
```

```
   STUDENT_ID     FIRST NAME
____________  __________________
          101 JOHN
          102 EMILY
          103 MICHAEL
          104 SARAH
          105 DAVID
          106 OLIVIA
          107 ETHAN
          108 SOPHIA
          109 AIDEN
          110 EMMA
          111 BENJAMIN
          112 MIA
          113 WILLIAM
          114 AVA
          115 JAMES

15 rows selected.
```

2. In a table of last names, you notice that some names have both leading and trailing spaces. Write an SQL query using the TRIM function to display the last names without leading and trailing spaces.

```
SELECT TRIM(BOTH ' ' FROM LAST_NAME) AS LastName FROM STUDENTINFO;

   LAST NAME
___________
SMITH
DAVIS
JOHNSON
WILSON
BROWN
LEE
MARTINEZ
TAYLOR
MILLER
ANDERSON
HARRIS
JOHNSON
WHITE
ROBINSON
TURNER

15 rows selected.
```

3.Retrieve a list of Student id from the STUDENTINFO table. Some Student id have both leading and trailing spaces. Use the TRIM function to clean the student id in your query.

```
SELECT TRIM(BOTH ' ' FROM STUDENT_ID) AS StudentID FROM STUDENTINFO;


   STUDENTID
____________
101
102
103
104
105
106
107
108
109
```

```
110
111
112
113
114
115

15 rows selected.
```

4. In a table containing phone numbers, some numbers have both leading and trailing spaces that need to be removed. Write an SQL query using the TRIM function to clean the phone number.

```
SELECT TRIM(BOTH ' ' FROM PHONE_NUMBER) AS PhoneNumber FROM
STUDENTINFO;


   PHONE NUMBER
______________
1234567890
9876543210
5551234567
7899876543
1112223333
4445556666
7778889999
2223334444
6667778888
3334445555
8889990000
9990001111
1112222
1112223333
2223334444

15 rows selected.
```

5. You have a table with Student names, and some names have both leading and trailing spaces. Write an SQL query using the TRIM function to display the Student names without any leading and trailing spaces.

SUBSTR Function:

1. Retrieve the first three characters of each student's first name from the STUDENTINFO table using the SUBSTR function. Display the STUDENT_ID and the extracted substrings.

```
SELECT STUDENT_ID, SUBSTR(FIRST_NAME, 1, 4) AS Extracted_Substring FROM
STUDENTINFO;


   STUDENT_ID    EXTRACTED_SUBSTRING
 -------------- ---------------------
           101 JOH
           102 EMI
           103 MIC
           104 SAR
           105 DAV
           106 OLI
           107 ETH
           108 SOP
           109 AID
           110 EMM
           111 BEN
           112 MIA
           113 WIL
           114 AVA
           115 JAM

15 rows selected.
```

2. You need to extract the last two characters from each student's last name. Write an SQL query using the SUBSTR function to retrieve the STUDENT_ID, LAST_NAME and the extracted substrings.

```
SELECT STUDENT_ID, LAST_NAME, SUBSTR(LAST_NAME , LENGTH(LAST_NAME ) -
1, 2) AS ExtractedSubstring FROM STUDENTINFO;
```

```
   STUDENT_ID     LAST_NAME  EXTRACT SUBSTRING

__ __________ ____________ _____________________
          101 SMITH        TH
          102 DAVIS        IS
          103 JOHNSON      ON
          104 WILSON       ON
          105 BROWN        WN
          106 LEE          EE
          107 MARTINEZ     EZ
          108 TAYLOR       OR
          109 MILLER       ER
          110 ANDERSON     ON
          111 HARRIS       IS
          112 JOHNSON      ON
          113 WHITE        TE
          114 ROBINSON     ON
          115 TURNER       ER

15 rows selected.
```

```
3.Display the STUDENT_ID, email, and only the domain part of each
student's email address using the SUBSTR function. Assume that the
domain is the character after the '@' symbol.

SELECT STUDENT_ID, EMAIL, SUBSTR(EMAIL, INSTR(EMAIL, '@') + 1) AS
Domain FROM STUDENTINFO;
```

```
   STUDENT_ID                        EMAIL       DOMAIN

_____________ ____________________________ ____________
          101 john.smith@email.com         email.com
          102 emily.davis@email.com        email.com
          103 michael.johnson@email.com    email.com
          104 sarah.wilson@email.com       email.com
          105 david.brown@email.com        email.com
          106 olivia.lee@email.com         email.com
          107 ethan.martinez@email.com     email.com
          108 sophia.taylor@email.com      email.com
          109 aiden.miller@email.com       email.com
          110 emma.anderson@email.com      email.com
```

```
    111 benjamin.harris@email.com     email.com
    112 mia.johnson@email.com         email.com
    113 willain.white@email.com       email.com
    114 ava.robinson@email.com        email.com
    115 james.turner@email.com        email.com

15 rows selected.
```

4.Calculate the length of the first five characters in each student's email address. Retrieve the STUDENT_ID, email, and the length of the substrings using the SUBSTR and LENGTH functions.

```
SELECT STUDENT_ID, EMAIL, LENGTH(SUBSTR(EMAIL, 1, 5)) AS
SubstringLength FROM STUDENTINFO;
```

```
   STUDENT_ID                         EMAIL    SUBSTRING LENGTH

_____________ ____________________________ __________________
          101 john.smith@email.com                          5
          102 emily.davis@email.com                         5
          103 michael.johnson@email.com                     5
          104 sarah.wilson@email.com                        5
          105 david.brown@email.com                         5
          106 olivia.lee@email.com                          5
          107 ethan.martinez@email.com                      5
          108 sophia.taylor@email.com                       5
          109 aiden.miller@email.com                        5
          110 emma.anderson@email.com                       5
          111 benjamin.harris@email.com                     5
          112 mia.johnson@email.com                         5
          113 willain.white@email.com                       5
          114 ava.robinson@email.com                        5
          115 james.turner@email.com                        5

15 rows selected.
```

5.Retrieve the STUDENT_ID, first_name, and the third to fifth characters of each students first name using the SUBSTR function.

```
SELECT STUDENT_ID, first_name, SUBSTR(first_name, 3, 3) AS
ExtractedSubstring FROM STUDENTINFO;
```

```
   STUDENT_ID     FIRST_NAME     EXTRACT SUBSTRING
 -------------  -------------  ---------------------
          101 JOHN             HN
          102 EMILY            ILY
          103 MICHAEL          CHA
          104 SARAH            RAH
          105 DAVID            VID
          106 OLIVIA           IVI
          107 ETHAN            HAN
          108 SOPHIA           PHI
          109 AIDEN            DEN
          110 EMMA             MA
          111 BENJAMIN         NJA
          112 MIA              A
          113 WILLIAM          LLI
          114 AVA              A
          115 JAMES            MES

15 rows selected.
```

```
------------------------------------------------------------------------------------------------------------------------
---
```

NVL Function:-

1. In the STUDENTINFO table, some students have missing values for their phone numbers (NULL). Use the NVL function to display 'Not Available' for students with no phone number. Retrieve the STUDENT_ID, first_name, and phone numbers.

```
SELECT STUDENT_ID, FIRST_NAME, NVL(phone_number, 'Not Available') AS
Phone FROM STUDENTINFO;
```

```
   STUDENT_ID     FIRST_NAME         PHONE
 -------------  -------------  -------------
```

```
      101 JOHN           1234567890
      102 EMILY          9876543210
      103 MICHAEL        5551234567
      104 SARAH          7899876543
      105 DAVID          1112223333
      106 OLIVIA         4445556666
      107 ETHAN          7778889999
      108 SOPHIA         2223334444
      109 AIDEN          6667778888
      110 EMMA           3334445555
      111 BENJAMIN       8889990000
      112 MIA            9990001111
      113 WILLIAM        1112222
      114 AVA            1112223333
      115 JAMES          2223334444

15 rows selected.
```

2. Calculate the age of each student based on their date of birth, and for students with missing birthdates (NULL), display 'Age Unknown' using the NVL function. Retrieve the STUDENT_ID, first_name, and the calculated age.

3. You want to categorise students as 'Male' or 'Female' based on their gender, but some have NULL values. Use the NVL function to categorise students with NULL gender values as 'Unknown'. Retrieve the STUDENT_ID, first_name, and the categorised gender.

```
 SELECT STUDENT_ID, FIRST_NAME,
         NVL(
              CASE
                 WHEN GENDER = 'Male' THEN 'Male'
                  WHEN GENDER = 'Female' THEN 'Female'
                  ELSE 'Unknown'
               END, 'Unknown') AS CategorizedGender
    FROM STUDENTINFO;
```

```
   STUDENT_ID     FIRST_NAME     CATEGORIZED GENDER

_____________  _____________  ____________________
          101  JOHN           Unknown
```

```
       102 EMILY              Unknown
       103 MICHAEL            Unknown
       104 SARAH              Unknown
       105 DAVID              Unknown
       106 OLIVIA             Unknown
       107 ETHAN              Unknown
       108 SOPHIA             Unknown
       109 AIDEN              Unknown
       110 EMMA               Unknown
       111 BENJAMIN           Unknown
       112 MIA                Unknown
       113 WILLIAM            Unknown
       114 AVA                Unknown
       115 JAMES              Unknown

15 rows selected.
```

4.Display the STUDENT_ID, email, and for students with NULL email addresses, show 'No Email' using the NVL function.

```
 SELECT STUDENT_ID, NVL(email, 'No Email') AS Email FROM STUDENTINFO;

   STUDENT_ID                          EMAIL
 _____________ ____________________________
          101 john.smith@email.com
          102 emily.davis@email.com
          103 michael.johnson@email.com
          104 sarah.wilson@email.com
          105 david.brown@email.com
          106 olivia.lee@email.com
          107 ethan.martinez@email.com
          108 sophia.taylor@email.com
          109 aiden.miller@email.com
          110 emma.anderson@email.com
          111 benjamin.harris@email.com
          112 mia.johnson@email.com
          113 willain.white@email.com
          114 ava.robinson@email.com
          115 james.turner@email.com
```

```
15 rows selected.
```

5. You have a table that date_of_birth, and some dates are missing (NULL). Use the NVL function to display 'date_of_birth Not Available' for student with null dates. Retrieve the student name and date.

```
SELECT FIRST_NAME, NVL(TO_CHAR(DATE_OF_BIRTH , 'DD-MM-YYYY'), 'Date of
Birth Not Available') AS DateOfBirth FROM STUDENTINFO;
```

```
   FIRST_NAME    DATE OF BIRTH

_____________  ______________
JOHN           15-05-1998
EMILY          20-03-1999
MICHAEL        10-07-1997
SARAH          05-01-2000
DAVID          30-09-1996
OLIVIA         18-12-1999
ETHAN          25-11-1998
SOPHIA         14-02-2002
AIDEN          12-04-1997
EMMA           07-08-2002
BENJAMIN       22-06-1995
MIA            01-10-1998
WILLIAM        12-03-2003
AVA            28-04-1999
JAMES          05-12-1996

15 rows selected.
```

NVL2 Function:-

1. You want to calculate the age for a student.Otherwise, its 0. Use the NVL2 function to calculate the bonus. Retrieve the STUDENT_ID, student_name, age, and the calculated age.

```
SELECT
    STUDENT_ID,
    FIRST_NAME,
    NVL2(date_of_birth, TRUNC(MONTHS_BETWEEN(SYSDATE, date_of_birth) /
12), 0) AS age,
    NVL2(date_of_birth, TRUNC(MONTHS_BETWEEN(SYSDATE, date_of_birth) /
12) * 100, 0) AS bonus
FROM STUDENTINFO;
```

```
   STUDENT_ID     FIRST_NAME     AGE     BONUS
_____________  _____________  ______  ________
          101 JOHN                25      2500
          102 EMILY               24      2400
          103 MICHAEL             26      2600
          104 SARAH               23      2300
          105 DAVID               27      2700
          106 OLIVIA              23      2300
          107 ETHAN               24      2400
          108 SOPHIA              21      2100
          109 AIDEN               26      2600
          110 EMMA                21      2100
          111 BENJAMIN            28      2800
          112 MIA                 25      2500
          113 WILLIAM             20      2000
          114 AVA                 24      2400
          115 JAMES               26      2600

15 rows selected.
```

2. In a table that last name, some quantities are missing (NULL). Use the NVL2 function to calculate the . If the quantity is missing, assume its 0. Retrieve the order IDs and adjusted quantities.

```
SELECT
    STUDENT_ID,
    NVL2(FIRST_NAME,LAST_NAME
    , 0) AS adjusted_quantity
FRO STUDENTINFO;
```

```
   STUDENT_ID     ADJUSTED_QUANTITY
_____________  ____________________
          101 SMITH
```

```
       102 DAVIS
       103 JOHNSON
       104 WILSON
       105 BROWN
       106 LEE
       107 MARTINEZ
       108 TAYLOR
       109 MILLER
       110 ANDERSON
       111 HARRIS
       112 JOHNSON
       113 WHITE
       114 ROBINSON
       115 TURNER

15 rows selected.
```

3. You need to categorise students based on their age. If a student is 18 or older, categorise them as 'Adult'; otherwise, categorise them as 'Minor.' Use the NVL2 function to categorise students with NULL birthdates as 'Unknown.' Retrieve the STUDENT_ID, first_name, and the categorised age.

```
SELECT STUDENT_ID, first_name,
  NVL2(DATE_OF_BIRTH,
    CASE
      WHEN FLOOR(MONTHS_BETWEEN(SYSDATE, DATE_OF_BIRTH) / 12) >= 1 THEN
'Adult'
      ELSE 'Minor'
    END,
    'Unknown'
  ) AS CategorizedAge
FROM STUDENTINFO;
```

```
   STUDENT_ID     FIRST_NAME     CATEGORIZED AGE

-------------  -------------  -----------------
          101 JOHNSON        Adult
```

```
       105 DAVID          Adult
       106 OLIVIA         Adult
       107 ETHAN          Adult
       108 SOPHIA         Adult
       109 AIDEN          Adult
       110 EMMA           Adult
       111 BENJAMIN       Adult
       112 MIA            Adult
       113 WILLIAM        Adult
       114 AVA            Adult
       115 JAMES          Adult

15 rows select
```

4.Calculate the age for students. If a students age is missing (NULL), use the NVL2 function to assume their age is null Retrieve the STUDENT_ID, first_name, and the age.

```
SELECT STUDENT_ID, first_name,
       NVL2(DATE_OF_BIRTH,FLOOR(MONTHS_BETWEEN(SYSDATE, date_of_birth)
/ 12),NULL) AS Age FROM STUDENTINFO;


   STUDENT_ID    FIRST_NAME     AGE
 -------------- -------------- ------
          101 JOHN               25
          102 EMILY              24
          103 MICHAEL            26
          104 SARAH              23
          105 DAVID              27
          106 OLIVIA             23
          107 ETHAN              24
          108 SOPHIA             21
          109 AIDEN              26
          110 EMMA               21
          111 BENJAMIN           28
          112 MIA                25
          113 WILLIAM            20
          114 AVA                24
          115 JAMES              26
```

```
15 rows selected.
```

5.In a table storing gender, some values are missing (NULL). Use the NVL2 function to calculate the adjusted values. If the values is missing, assume its Not null. Retrieve the student's names and adjusted values.

```
SELECT FIRST_NAME, NVL2(GENDER, GENDER, 'Not Available') AS
AdjustedGender FROM STUDENTINFO;
```

```
   FIRST_NAME    ADJUSTED GENDER

_____________ _________________
JOHN          M
EMILY         M
MICHAEL       M
SARAH         F
DAVID         M
OLIVIA        F
ETHAN         M
SOPHIA        F
AIDEN         M
EMMA          F
BENJAMIN      M
MIA           F
WILLIAM       M
AVA           F
JAMES         M

15 rows selected.
```

LENGTH Function:-

1.Calculate the length of each students first name in the STUDENTINFO table. Retrieve the STUDENT_ID, first_name, and the length of the first names.

```
SELECT STUDENT_ID, FIRST_NAME, LENGTH(FIRST_NAME) AS FirstNameLength
FROM STUDENTINFO;


   STUDENT_ID     FIRST_NAME     FIRSTNAME LENGTH

 _____________  _____________  __________________
          101 JOHN                            4
          102 EMILY                           5
          103 MICHAEL                         7
          104 SARAH                           5
          105 DAVID                           5
          106 OLIVIA                          6
          107 ETHAN                           5
          108 SOPHIA                          6
          109 AIDEN                           5
          110 EMMA                            4
          111 BENJAMIN                        8
          112 MIA                             3
          113 WILLIAM                         7
          114 AVA                             3
          115 JAMES                           5

15 rows selected.
```

2. You have a table that stores email addresses, and you want to find the length of each email address. Retrieve the email addresses and their lengths using the LENGTH function.

```
SELECT EMAIL, LENGTH(EMAIL) AS EmailLength FROM STUDENTINFO;


                         EMAIL    LENGTH

 ____________________________  ______________
john.smith@email.com                    20
emily.davis@email.com                   21
michael.johnson@email.com               25
sarah.wilson@email.com                  22
david.brown@email.com                   21
olivia.lee@email.com                    20
ethan.martinez@email.com                24
```

```
sophia.taylor@email.com                  23
aiden.miller@email.com                   22
emma.anderson@email.com                  23
benjamin.harris@email.com                25
mia.johnson@email.com                    21
willain.white@email.com                  23
ava.robinson@email.com                   22
james.turner@email.com                   22

15 rows selected.
```

3. Determine the number of characters in each students last name in the STUDENTINFO table. Retrieve the STUDENT_ID, LAST_NAME and the length of the last names.

```
SELECT STUDENT_ID, LAST_NAME, LENGTH(LAST_NAME ) AS LastNameLength FROM
STUDENTINFO;

   STUDENT_ID    LAST_NAME  LASTNAMELENGTH


__ __________ ____________ _________________
          101 SMITH                       5
          102 DAVIS                       5
          103 JOHNSON                     7
          104 WILSON                      6
          105 BROWN                       5
          106 LEE                         3
          107 MARTINEZ                    8
          108 TAYLOR                      6
          109 MILLER                      6
          110 ANDERSON                    8
          111 HARRIS                      6
          112 JOHNSON                     7
          113 WHITE                       5
          114 ROBINSON                    8
          115 TURNER                      6

15 rows selected.
```

4.Calculate the length of each phone number in a table that number. Retrieve the phone number and their lengths using the LENGTH function.

```
SELECT PHONE_NUMBER, LENGTH(PHONE_NUMBER) AS PhoneNumberLength FROM
STUDENTINFO;
```

```
   PHONE_NUMBER     PHONE NUMBER LENGTH
_____________________________________
1234567890                        10
9876543210                        10
5551234567                        10
7899876543                        10
1112223333                        10
4445556666                        10
7778889999                        10
2223334444                        10
6667778888                        10
3334445555                        10
8889990000                        10
9990001111                        10
1112222                            7
1112223333                        10
2223334444                        10

15 rows selected.
```

5.You want to find the length of each Student’s full name in a table. Retrieve the First names,Last Name, Students Full Name and their lengths using the LENGTH function.

```
SELECT
    first_name,
    LAST_NAME
    ,
    CONCAT(first_name, LAST_NAME
    ) AS Full_Name,
    LENGTH(CONCAT(first_name,LAST_NAME
    )) AS Full_Name_Length
FROM STUDENTINFO;
```

```
   FIRST_NAME     LAST_NAME
            FULL_NAME     FULL_NAME_LENGTH

____________   ___________   ________________   ___________________
JOHN           SMITH         JOHNSMITH                            9
EMILY          DAVIS         EMILYDAVIS                          10
MICHAEL        JOHNSON       MICHAEL JOHNSON                       14
SARAH          WILSON        SARAH WILSON                          11
DAVID          BROWN         DAVID BROWN                           10
OLIVIA         LEE           OLIVIALEE                            9
ETHAN          MARTINEZ      ETHANMARTINEZ                       13
SOPHIA         TAYLOR        SOPHIATAYLOR                        12
AIDEN          MILLER        AIDENMILLER                         11
EMMA           ANDERSON      EMMAANDERSON                        12
BENJAMIN       HARRIS        BENJAMIN HARRIS                       14
MIA            JOHNSON       MIAJOHNSON                          10
WILLIAM        WHITE         WILLIAM WHITE                         12
AVA            ROBINSON      AVAROBINSON                         11
JAMES          TURNER        JAMES TURNER                          11

15 rows selected.
```

SOUNDEX Function STUDENTINFO Table):

1. You have a requirement to find students in the STUDENTINFO table who may have similar-sounding last names. Write an SQL query using theSoundex function to display the STUDENT_ID, LAST_NAME and Soundex codes for students with last names that sound alike.

```
 SELECT STUDENT_ID,FIRST_NAME,LENGTH(FIRST_NAME)AS NAME_LENGTH FROM
STUDENTINFO
```

```
   STUDENT_ID     FIRST_NAME     SOUNDEXCODE

_____________  _____________  ______________
          101 JOHN           J500
```

```
       102 EMILY         E540
       103 MICHAEL       M240
       104 SARAH         S600
       105 DAVID         D130
       106 OLIVIA        O410
       107 ETHAN         E350
       108 SOPHIA        S100
       109 AIDEN         A350
       110 EMMA          E500
       111 BENJAMIN      B525
       112 MIA           M000
       113 WILLIAM       W450
       114 AVA           A100
       115 JAMES         J520

15 rows selected.
```

2.Use the Soundex function in the STUDENTINFO table to calculate the Soundex codes for each student's last name. Retrieve the STUDENT_ID, LAST_NAMEand Soundex codes.

```
 SELECT STUDENT_ID, LAST_NAME, SOUNDEX(LAST_NAME) AS SoundexCode FROM
STUDENTINFO;


   STUDENT_ID    LAST_NAME SOUNDEXCODE

__ __________ ____________ ______________
       101 SMITH        S530
       102 DAVIS        D120
       103 JOHNSON      J525
       104 WILSON       W425
       105 BROWN        B650
       106 LEE          L000
       107 MARTINEZ     M635
       108 TAYLOR       T460
       109 MILLER       M460
       110 ANDERSON     A536
       111 HARRIS       H620
       112 JOHNSON      J525
       113 WHITE         W300
       114 ROBINSON      R152
       115 TURNER        T656

    15 rows selected.
```

3. In the STUDENTINFO table, some students may have last names that sound similar but are spelled differently. Write an SQL query with the Soundex function to identify such students. Display the STUDENT_ID, LAST_NAME , and Soundex codes.

```
SELECT SOUNDEX(FIRST_NAME),SOUNDEX(LAST_NAME),SOUNDEX(GENDER)FROM
STUDENTINFO;


   SOUNDEX(FIRST_NAME)     SOUNDEX(LAST_NAME )   SOUNDEX(GENDER)


__ ____________________  _____________________  __________________
J500                     S530                   M000
E540                     D120                   M000
M240                     J525                   M000
S600                     W425                   F000
D130                     B650                   M000
O410                     L000                   F000
E350                     M635                   M000
S100                     T460                   F000
A350                     M460                   M000
E500                     A536                   F000
B525                     H620                   M000
M000                     J525                   F000
W450                     W300                   M000
A100                     R152                   F000
J520                     T656                   M000

15 rows selected.
```

4. You are tasked with finding potential duplicate student records in the STUDENTINFO table based on similar-sounding last names. Write an SQL query using the Soundex function to retrieve the STUDENT_ID,

```
LAST_NAMEand Soundex codes for students with last names that sound
alike.
```