Name	ID	STUDENT SIGN
Sumaiya Tasnim	23-50014-1	Sumaiya Tasnim

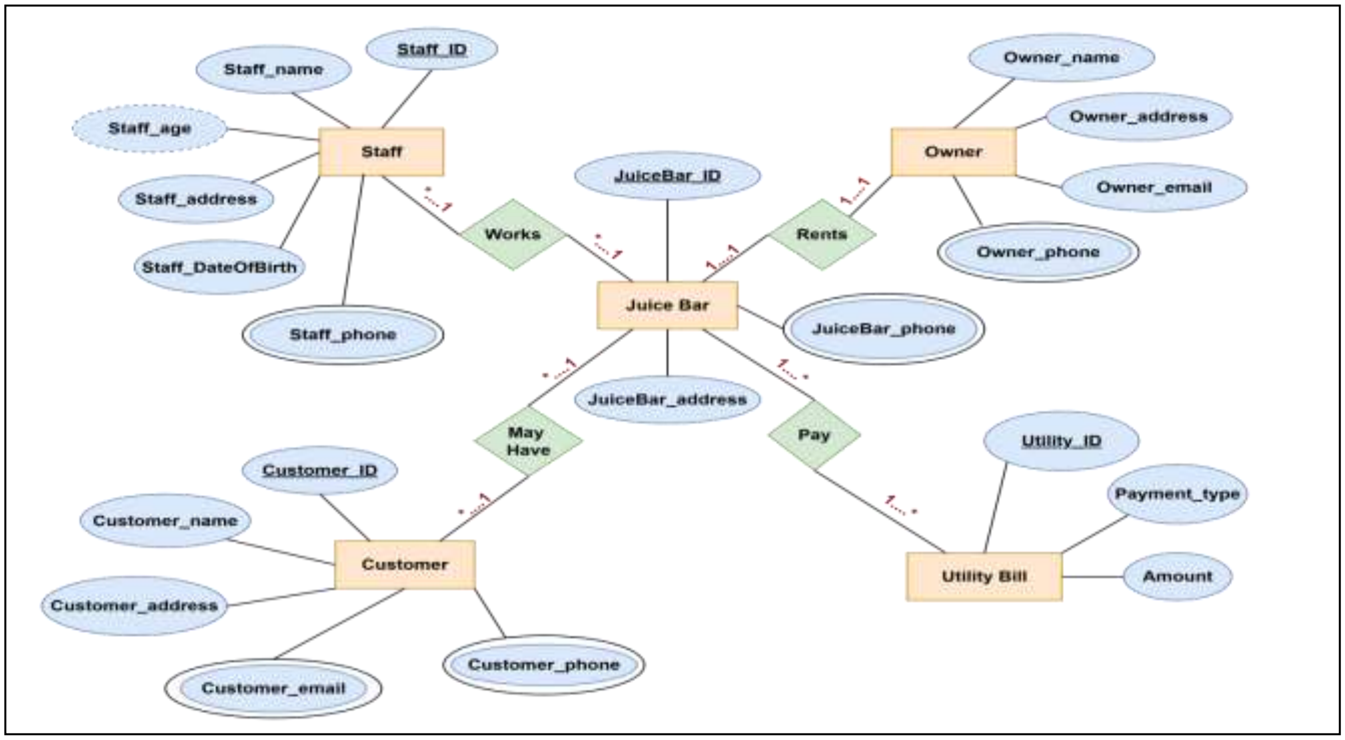
Instructions:

- **Make sure to write your Name, ID and Signature on this document.**
- **First write your signature on a paper then take photo of that signature and use it for signing this document.**
- **After completing the requirements of the midterm assignment by editing this document, upload this document in the link provided in your VUES Student Account.**
- **Submission Deadline: 18th August 2025, 11:59pm.**

Midterm Assignment

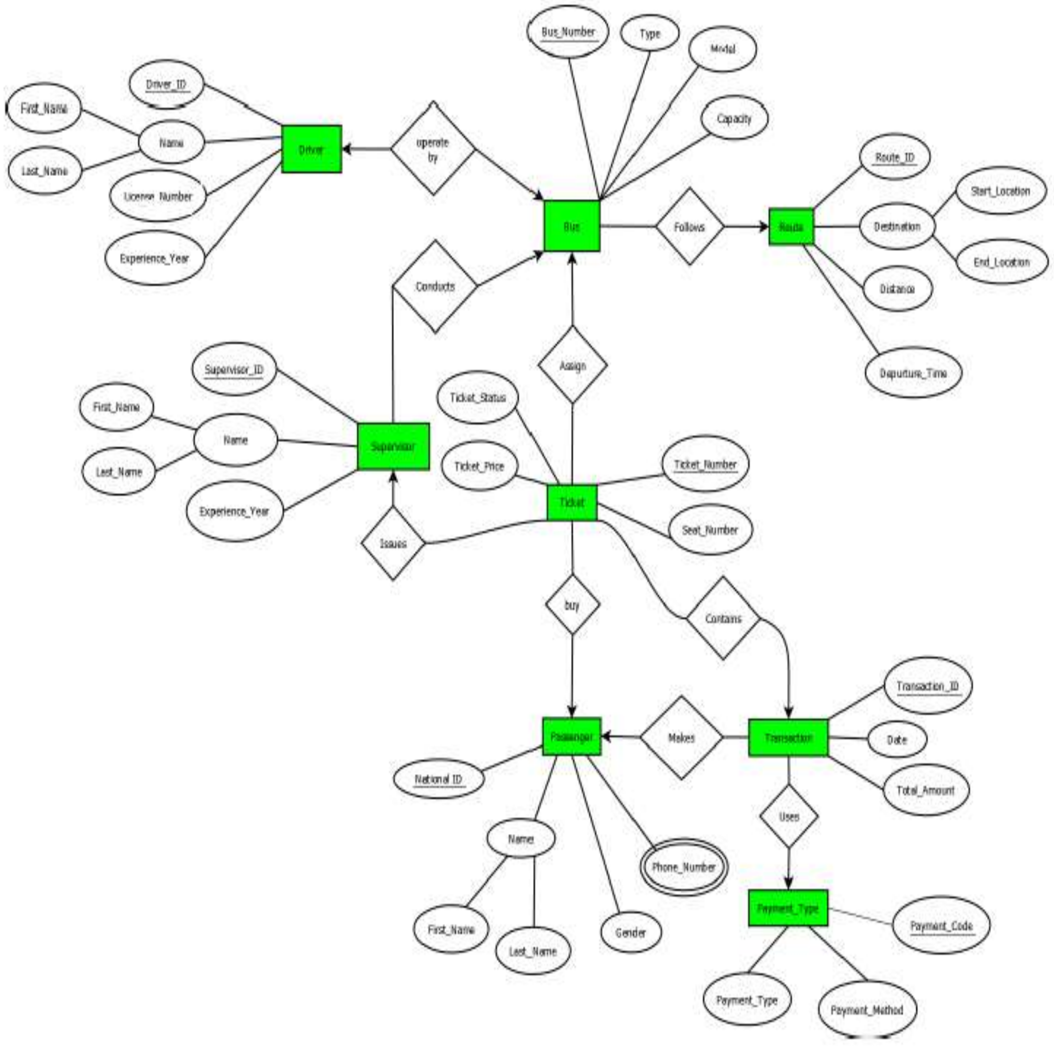
1. **Below a scenario has been given draw the ER Diagram.**
   Draw with proper annotations (use DIA, VISIO, MS WORD etc.).
   For reference see ERDiagramTutorial.
   In a Juice Bar Management System, one Juice Bar may have many staff. But one staff can work in one Juice Bar only. Each staff has a unique identification number, name, age, address, date of birth and phone number. Juice Bar has a unique identification number, address, and phone. One Juice Bar maybe rented by exactly one owner. One owner may rent exactly one Juice Bar. Owner is defined by name, address, email and phone. A Juice Bar may have many customers. Each customer has a unique identification number, name, address, email, phone. Each customer can have more than one email address and phone number. A Juice Bar must pay the utility bill which has a unique identification number, payment_type and amount.

Answer Box 1: (I have used "draw.io" as tool to create the ER diagram)



2. **Below an ER Diagram has been given write the scenario.**

For reference see ERDiagramTutorial.

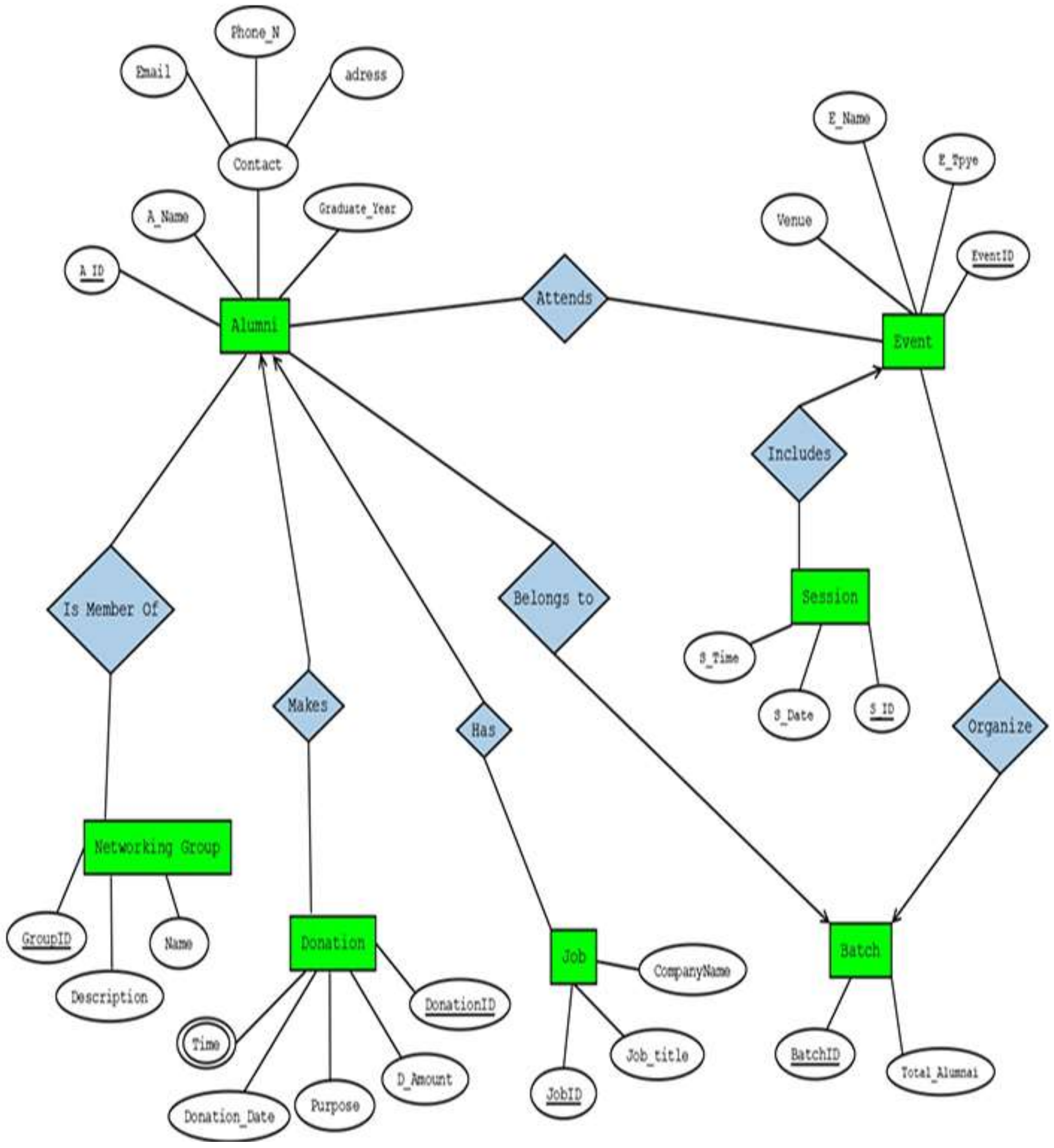


Answer Box 2:

In a Bus Ticketing Management System, one **bus** is operated by exactly one **driver**, while one **driver** can operate only one **bus**. Each driver has a unique driver ID, name, license number, first name, last name, and years of experience. Each bus has a unique bus number and is defined by its type, model, and capacity. A **bus** follows exactly one **route**, and a **route** can be followed by buses. A **route** has a unique route ID and is defined by start location, end location, destination, distance, and departure time.Each **bus** is conducted by exactly one **supervisor**, and one supervisor can conduct only one bus. A **supervisor** is uniquely identified by supervisor ID and is described by name, first name, last name, and experience year. A **supervisor** issues many **tickets**, but each **ticket** is issued by only one supervisor. A **ticket** is assigned to exactly one bus and contains details such as ticket number, ticket price, ticket status, and seat number.A **ticket** is bought by one or more **passengers**, and each **passenger** can buy multiple **tickets**. Each passenger is uniquely identified by their national ID and is described by name, gender, phone number, first name, and last name.A **passenger** makes one or more **transactions**, and each transaction is made by one passenger only. A **transaction** is uniquely identified by a transaction ID and is described by date and ticket amount. Each transaction uses exactly one **payment type**, but a payment type can be used in many transactions. A **payment type** is defined by its payment code, method, and type.

3. **Normalize the ER Diagram given below up to 3rd Normal Form and finalize the tables that needs to be created. Then (in Oracle using SQL) write down the queries that are required to create all the tables with necessary constraints. Also insert at least 3 rows of data in each created table.**

   For reference see NormalizationTutorial and BasicSQLTutorial.



Answer Box 3 (Normalization steps in detail as shown in Normalization Tutorial Slide + all the queries required to create the tables and insert data after Normalization):

Attends

UNF

Alumni(A_ID ,A_name, Contact, address, Phone_N, Email, Graduate_Year, Venue, E_Name, E_Type, Event_ID)

1NF

There is no multi valued attribute. Relation is already in 1NF.

1. A_ID ,A_name, Contact, address, Phone_N, Email, Graduate_Year, Venue, E_Name, E_Type, Event_ID

2NF

1. A_ID ,A_name, Contact, address, Phone_N, Email, Graduate_Year

2. Venue, E_Name, E_Type, Event_ID

3NF

There is no transitive dependency. Relation already in 3NF.

1. A_ID ,A_name, Contact, address, Phone_N, Email, Graduate_Year

2. Venue, E_Name, E_Type, Event_ID

Table Creation

1. A_ID ,A_name, Contact, address, Phone_N, Email, Graduate_Year

2. Venue, E_Name, E_Type, Event_ID, **A_ID**

Is Member Of

UNF

Alumni(A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, GroupID, Group_Name, Group_Description)

1NF

There is no multi-valued attribute. Relation is already in 1NF.

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, GroupID, Group_Name, Group_Description

2NF

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year

2. GroupID, Group_Name, Group_Description

3NF
There is no transitive dependency. Relation already in 3NF.

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year

2. GroupID, Group_Name, Group_Description

Table Creation

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year

2. GroupID, Group_Name, Group_Description, **A_ID**

Makes

UNF
Alumni(A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, DonationID, Donation_Date, Time, Purpose, D_Amount)

1NF
Time is a multi valued attribute.

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, DonationID, Donation_Date, Time, Purpose, D_Amount

2NF

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year

2. DonationID, Donation_Date, Time, Purpose, D_Amount

3NF
There is no transitive dependency. Relation already in 3NF.

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year
2. DonationID, Donation_Date, Time, Purpose, D_Amount

Table Creation

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year

2. DonationID, Donation_Date, Time, Purpose, D_Amount, **A_ID**

Has

UNF
Alumni(A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, JobID, Job_Title, CompanyName)

1NF
There is no multi-valued attribute. Relation is already in 1NF.

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, JobID, Job_Title, CompanyName

2NF

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year
2. JobID, Job_Title, CompanyName

3NF
There is no transitive dependency. Relation already in 3NF.

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year
2. JobID, Job_Title, CompanyName

Table Creation

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year
2. JobID, Job_Title, CompanyName, **A_ID**

Belongs To

UNF
Alumni(A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, BatchID, Total_Alumni)

1NF
There is no multi-valued attribute. Relation is already in 1NF.

1.A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, BatchID, Total_Alumni

2NF

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year
2. BatchID, Total_Alumni

3NF
There is no transitive dependency. Relation already in 3NF.

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year
2. BatchID, Total_Alumni

Table Creation

1. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, BatchID
2. BatchID, Total_Alumni, **A_ID**

Includes

UNF
Event(EventID, E_Name, E_Type, Venue, S_ID, S_Date, S_Time)

1NF
There is no multi-valued attribute. Relation is already in 1NF.

1. EventID, E_Name, E_Type, Venue, S_ID, S_Date, S_Time

2NF

1. EventID, E_Name, E_Type, Venue
2. S_ID, S_Date, S_Time

3NF
There is no transitive dependency. Relation already in 3NF.

1. EventID, E_Name, E_Type, Venue
2. S_ID, S_Date, S_Time

Table Creation

1. EventID, E_Name, E_Type, Venue
2. S_ID, S_Date, S_Time, **EventID**

Organize

UNF
Event(EventID, E_Name, E_Type, Venue, BatchID, Total_Alumni)

1NF
There is no multi-valued attribute. Relation is already in 1NF.

1. EventID, E_Name, E_Type, Venue, BatchID, Total_Alumni

2NF

1. EventID, E_Name, E_Type, Venue
2. BatchID, Total_Alumni

3NF
There is no transitive dependency. Relation already in 3NF.

1. EventID, E_Name, E_Type, Venue
2. BatchID, Total_Alumni

Table Creation

1. EventID, E_Name, E_Type, Venue
2. BatchID, Total_Alumni, **EventID**

Temporary Tables

1. A_ID ,A_name, Contact, address, Phone_N, Email, Graduate_Year

2. Venue, E_Name, E_Type, Event_ID, **A_ID**

~~3. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year~~

4. GroupID, Group_Name, Group_Description, **A_ID**

~~5. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year~~

6. DonationID, Donation_Date, Time, Purpose, D_Amount, **A_ID**

~~7. A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year~~

8. JobID, Job_Title, CompanyName, **A_ID**

9. ~~A_ID, A_Name, Contact, Address, Phone_N, Email, Graduate_Year, BatchID~~

10. BatchID, Total_Alumni, **A_ID**

~~11. EventID, E_Name, E_Type, Venue~~

12. S_ID, S_Date, S_Time, **EventID**

~~13. EventID, E_Name, E_Type, Venue~~

~~14.BatchID, Total_Alumni, **EventID**~~

Final Tables

1. A_ID ,A_name, Contact, address, Phone_N, Email, Graduate_Year

2. Venue, E_Name, E_Type, Event_ID, **A_ID**

3. GroupID, Group_Name, Group_Description, **A_ID**

4. DonationID, Donation_Date, Time, Purpose, D_Amount, **A_ID**

5. JobID, Job_Title, CompanyName, **A_ID**

6. BatchID, Total_Alumni, **A_ID**

7. S_ID, S_Date, S_Time, **EventID**

(In Oracle using SQL) writing the queries below that are required to create all the tables with necessary constraints :

```
-- 1. Alumni Table

CREATE TABLE Alumni (

   A_ID NUMBER PRIMARY KEY,

   A_name VARCHAR2(100) NOT NULL,

   Contact VARCHAR2(50),

   Address VARCHAR2(200),

   Phone_N VARCHAR2(15) UNIQUE,

   Email VARCHAR2(100) UNIQUE,

   Graduate_Year NUMBER(4)     );

-- 2. Event Table

CREATE TABLE Event (

   Event_ID NUMBER PRIMARY KEY,

   Venue VARCHAR2(100),

   E_Name VARCHAR2(100),

   E_Type VARCHAR2(50),

   A_ID NUMBER,

   FOREIGN KEY (A_ID) REFERENCES Alumni(A_ID)

);
```

```sql
-- 3. Networking Group Table

CREATE TABLE Networking_Group (

   GroupID NUMBER PRIMARY KEY,

   Group_Name VARCHAR2(100),

   Group_Description VARCHAR2(200),

   A_ID NUMBER,

   FOREIGN KEY (A_ID) REFERENCES Alumni(A_ID)

);

-- 4. Donation Table

CREATE TABLE Donation (

   DonationID NUMBER PRIMARY KEY,

   Donation_Date DATE,

   Time VARCHAR2(10),

   Purpose VARCHAR2(200),

   D_Amount NUMBER(10,2),

   A_ID NUMBER,

   FOREIGN KEY (A_ID) REFERENCES Alumni(A_ID)

);

-- 5. Job Table

CREATE TABLE Job (

   JobID NUMBER PRIMARY KEY,

   Job_Title VARCHAR2(100),

   CompanyName VARCHAR2(100),

   A_ID NUMBER,

   FOREIGN KEY (A_ID) REFERENCES Alumni(A_ID)

);
```

-- 6. Batch Table

```
CREATE TABLE Batch (

    BatchID NUMBER PRIMARY KEY,

    Total_Alumni NUMBER,

    A_ID NUMBER,

    FOREIGN KEY (A_ID) REFERENCES Alumni(A_ID)

);
```

-- 7. Session Table

```
CREATE TABLE Event_Session (

    S_ID NUMBER PRIMARY KEY,

    S_Date DATE,

    S_Time VARCHAR2(10),

    EventID NUMBER,

    FOREIGN KEY (EventID) REFERENCES Event(Event_ID)

);
```

Inserting at least 3 rows of data in each created table:

-- Alumni

```
INSERT INTO Alumni VALUES (1, 'John Smith', 'Facebook', 'NY, USA', '1234567890', 'john@example.com', 2015);

INSERT INTO Alumni VALUES (2, 'Sara Khan', 'Twitter', 'London, UK', '2345678901', 'sara@example.com', 2016);

INSERT INTO Alumni VALUES (3, 'David Lee', 'LinkedIn', 'Toronto, Canada', '3456789012', 'david@example.com',
2017);
```

-- Event

```
INSERT INTO Event VALUES (101, 'Auditorium', 'Alumni Meet', 'Social', 1);

INSERT INTO Event VALUES (102, 'Hall B', 'Career Fair', 'Professional', 2);

INSERT INTO Event VALUES (103, 'Conference Room', 'Workshop', 'Educational', 3);
```

-- Networking Group

```
INSERT INTO Networking_Group VALUES (201, 'Tech Group', 'Focus on IT networking', 1);

INSERT INTO Networking_Group VALUES (202, 'Business Leaders', 'Entrepreneurship and startups', 2);

INSERT INTO Networking_Group VALUES (203, 'Research Circle', 'Scientific research collaboration', 3);
```

```sql
-- Donation

INSERT INTO Donation VALUES (301, TO_DATE('2024-05-10','YYYY-MM-DD'), '10:00AM', 'Library Renovation', 5000, 1);

INSERT INTO Donation VALUES (302, TO_DATE('2024-06-15','YYYY-MM-DD'), '02:00PM', 'Scholarship Fund', 3000, 2);

INSERT INTO Donation VALUES (303, TO_DATE('2024-07-20','YYYY-MM-DD'), '11:00AM', 'Sports Complex', 7000, 3);

-- Job

INSERT INTO Job VALUES (401, 'Software Engineer', 'Google', 1);

INSERT INTO Job VALUES (402, 'Marketing Manager', 'Amazon', 2);

INSERT INTO Job VALUES (403, 'Data Scientist', 'Microsoft', 3);

-- Batch

INSERT INTO Batch VALUES (501, 150, 1);

INSERT INTO Batch VALUES (502, 200, 2);

INSERT INTO Batch VALUES (503, 180, 3);

-- Session

INSERT INTO Event_Session VALUES (601, TO_DATE('2024-08-01','YYYY-MM-DD'), '09:00AM', 101);

INSERT INTO Event_Session VALUES (602, TO_DATE('2024-08-05','YYYY-MM-DD'), '01:00PM', 102);

INSERT INTO Event_Session VALUES (603, TO_DATE('2024-08-10','YYYY-MM-DD'), '03:00PM', 103);
```

4.Query Writing (continuation of Question 3) (Write down the question and the answer. Give full screenshot of the Oracle 10g Homepage that contains the answer and result)

-All screenshots MUST include the DATE and TIME feature from the screen of the machine (PC, Laptop etc.) used

SQL

-2 single-row function

-2 group function

-2 subquery

-2 joining

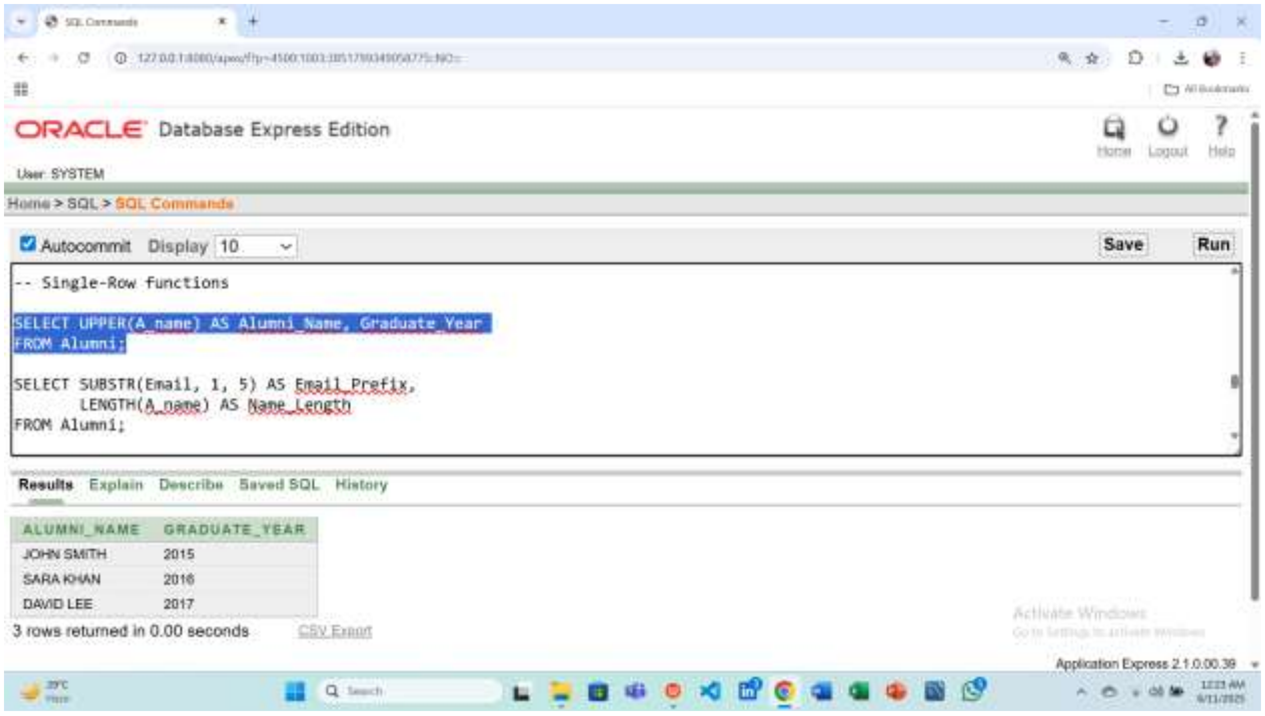
For reference see BasicSQLTutorial and AdvanceSQLTutorial.

Answer Box 4:

1. Single-Row Functions

Question(1): Display the names of alumni in uppercase and show their graduation year.
Answer:

SELECT UPPER(A_name) AS Alumni_Name, Graduate_Year FROM Alumni;



```
-- Single-Row functions

SELECT UPPER(A_name) AS Alumni_Name, Graduate_Year
FROM Alumni;

SELECT SUBSTR(Email, 1, 5) AS Email_Prefix,
       LENGTH(A_name) AS Name_Length
FROM Alumni;
```

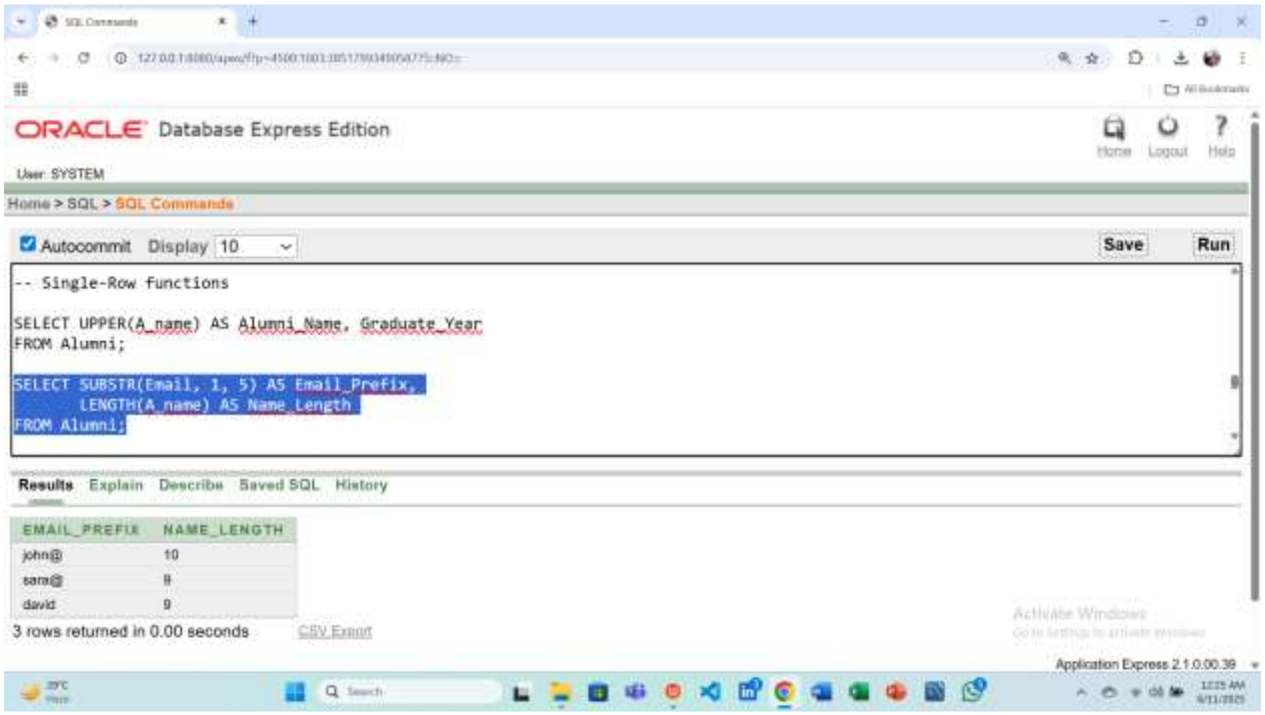
ALUMNI_NAME	GRADUATE_YEAR
JOHN SMITH	2015
SARA KHAN	2016
DAVID LEE	2017

3 rows returned in 0.00 seconds

Question(2): Display the first 5 characters of each alumni's email and the length of their name.
Answer:

SELECT SUBSTR(Email, 1, 5) AS Email_Prefix,

LENGTH(A_name) AS Name_LengthFROM Alumni;



```
-- Single-Row functions

SELECT UPPER(A_name) AS Alumni_Name, Graduate_Year
FROM Alumni;

SELECT SUBSTR(Email, 1, 5) AS Email_Prefix,
       LENGTH(A_name) AS Name_Length
FROM Alumni;
```

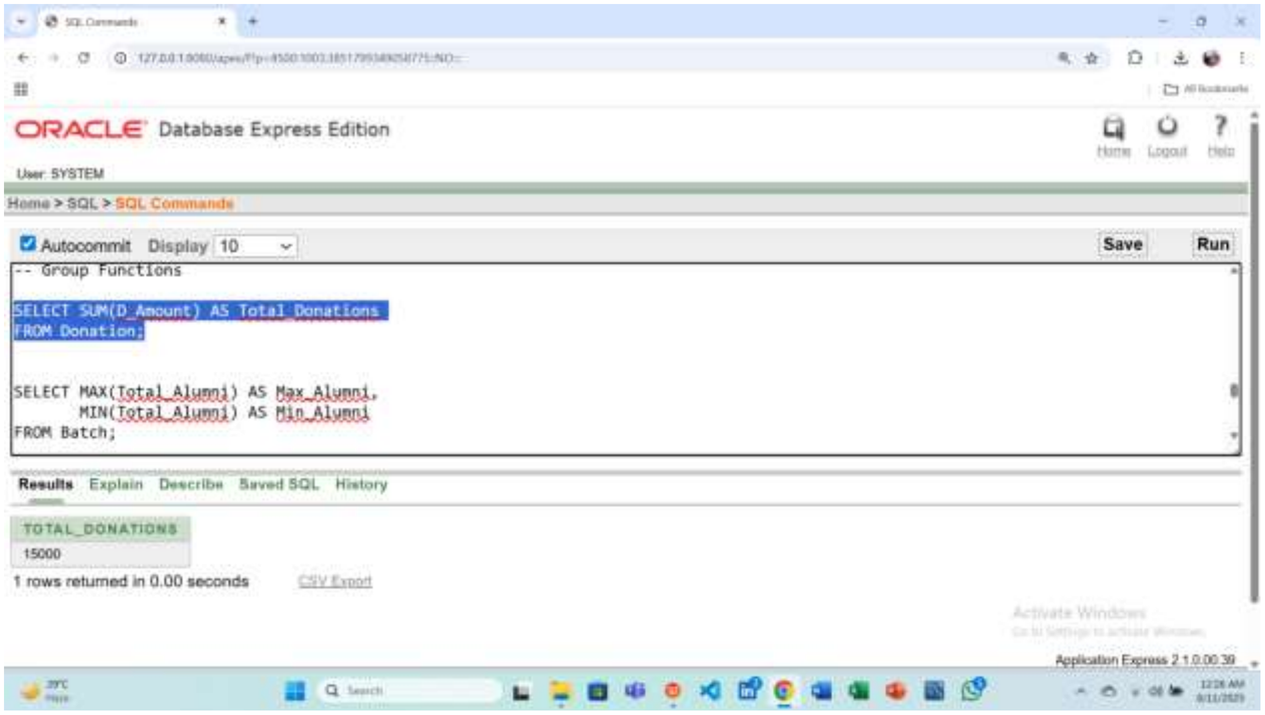
EMAIL_PREFIX	NAME_LENGTH
john@	10
sara@	9
david	9

3 rows returned in 0.00 seconds

2. Group Functions

Question(1): Find the total donation amount received from all alumni.
Answer:

SELECT SUM(D_Amount) AS Total_DonationsFROM Donation;



```
-- Group Functions

SELECT SUM(D_Amount) AS Total_Donations
FROM Donation;


SELECT MAX(Total_Alumni) AS Max_Alumni,
       MIN(Total_Alumni) AS Min_Alumni
FROM Batch;
```

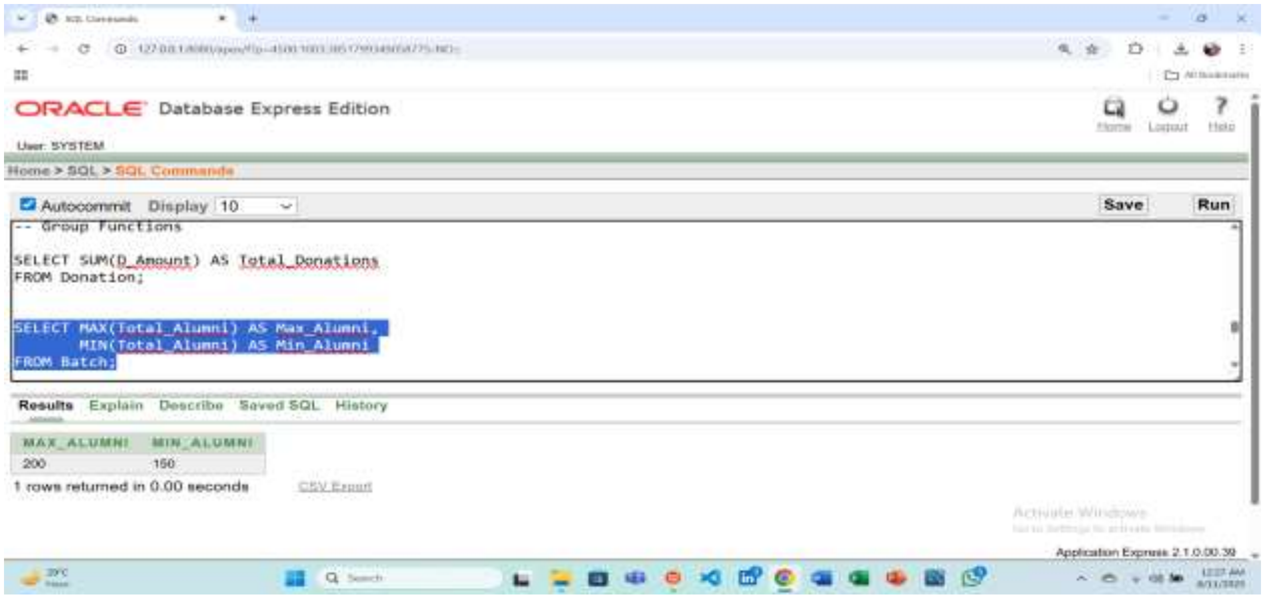
TOTAL_DONATIONS
15000

1 rows returned in 0.00 seconds

Question(2): Find the maximum and minimum total alumni count among all batches.
Answer:

SELECT MAX(Total_Alumni) AS Max_Alumni,

MIN(Total_Alumni) AS Min_AlumniFROM Batch;



ORACLE Database Express Edition

User: SYSTEM

Home > SQL > SQL Commands

Autocommit Display 10

```
-- Group Functions

SELECT SUM(D_Amount) AS Total_Donations
FROM Donation;


SELECT MAX(Total_Alumni) AS Max_Alumni,
       MIN(Total_Alumni) AS Min_Alumni
FROM Batch;
```

Results Explain Describe Saved SQL History

```
MAX_ALUMNI   MIN_ALUMNI
200          150
1 rows returned in 0.00 seconds      CSV Export
```

Application Express 2.1.0.00.39

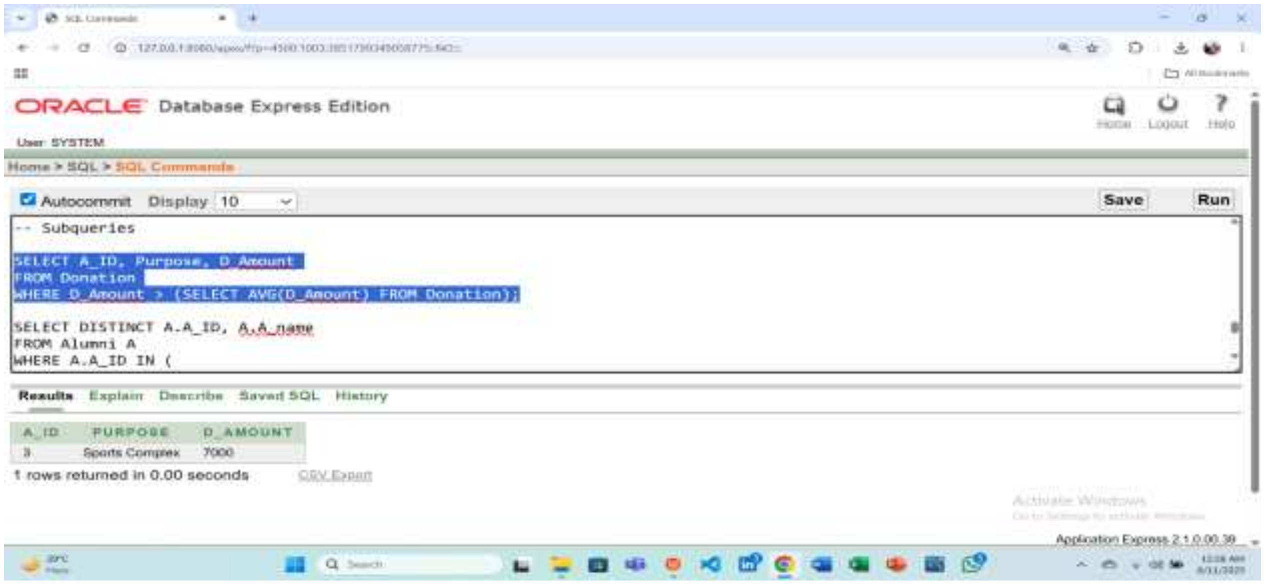
3. Subqueries

Question(1): Display the alumni who donated more than the average donation amount.
Answer:

SELECT A_ID, Purpose, D_Amount

FROM Donation

WHERE D_Amount > (SELECT AVG(D_Amount) FROM Donation);



ORACLE Database Express Edition

User: SYSTEM

Home > SQL > SQL Commands

Autocommit Display 10

```
-- Subqueries

SELECT A_ID, Purpose, D_Amount
FROM Donation
WHERE D_Amount > (SELECT AVG(D_Amount) FROM Donation);

SELECT DISTINCT A.A_ID, A.A_name
FROM Alumni A
WHERE A.A_ID IN (
```

Results Explain Describe Saved SQL History

```
A_ID   PURPOSE          D_AMOUNT
3      Sports Complex   7000
1 rows returned in 0.00 seconds      CSV Export
```

Application Express 2.1.0.00.39

Question(2): Display alumni who participated in events with the same venue as event ID 101.
Answer:

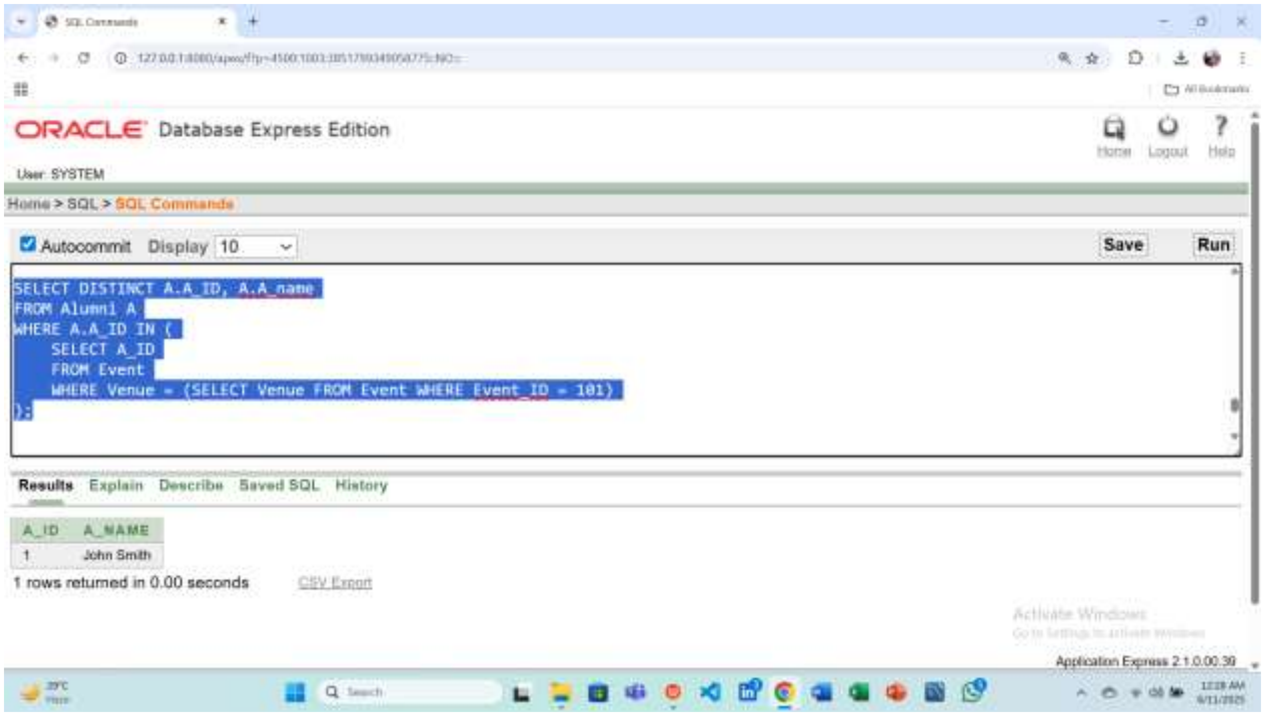
SELECT DISTINCT A.A_ID, A.A_nameFROM Alumni AWHERE A.A_ID IN (

SELECT A_ID

FROM Event

WHERE Venue = (SELECT Venue FROM Event WHERE Event_ID = 101)

);



ORACLE Database Express Edition
User: SYSTEM
Home > SQL > SQL Commands
Autocommit Display 10
Save
Run
Results Explain Describe Saved SQL History

```
SELECT DISTINCT A.A_ID, A.A_name
FROM Alumni A
WHERE A.A_ID IN (
    SELECT A_ID
    FROM Event
    WHERE Venue = (SELECT Venue FROM Event WHERE Event_ID = 101)
);
```

```
A_ID  A_NAME
1     John Smith
1 rows returned in 0.00 seconds
```

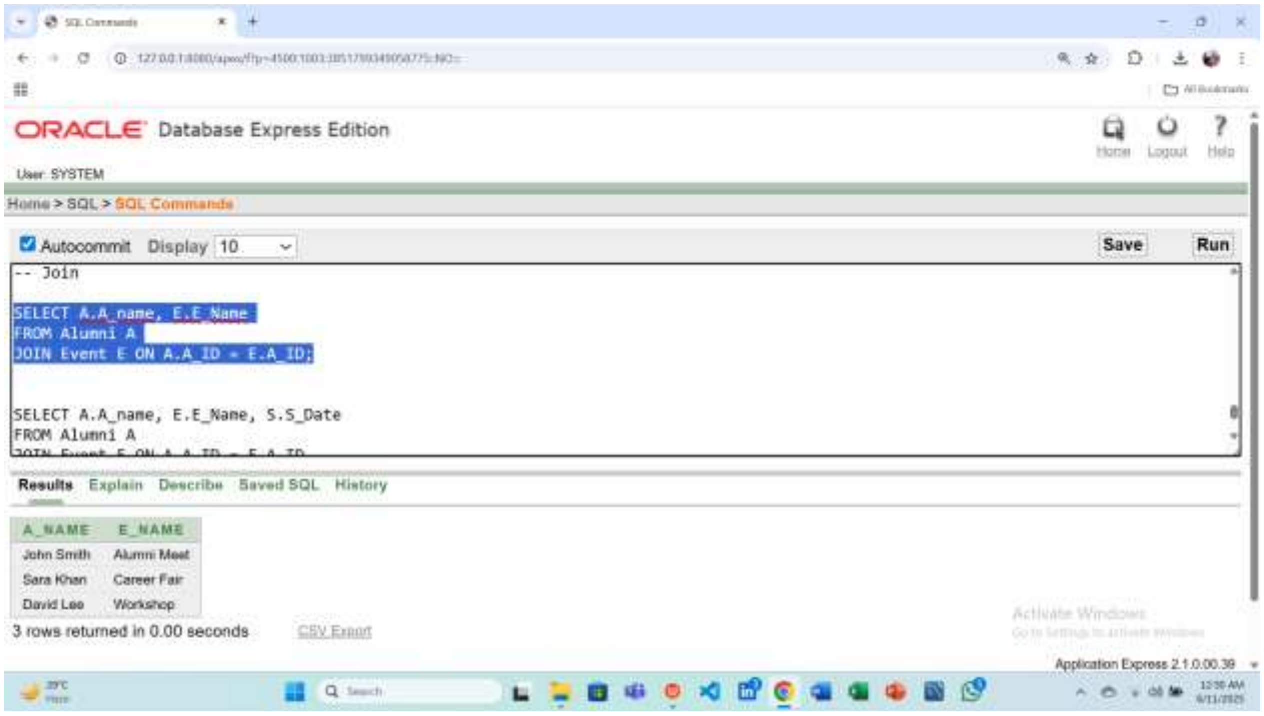
4. Joins

Question(1): Display all alumni names along with the event names they are associated with.
Answer:

SELECT A.A_name, E.E_Name

FROM Alumni A

JOIN Event E ON A.A_ID = E.A_ID;


ORACLE Database Express Edition
User: SYSTEM
Home > SQL > SQL Commands
Autocommit Display 10
Save
Run
Results Explain Describe Saved SQL History
CSV Export

```
-- Join

SELECT A.A_name, E.E_Name
FROM Alumni A
JOIN Event E ON A.A_ID = E.A_ID;


SELECT A.A_name, E.E_Name, S.S_Date
FROM Alumni A
```

A_NAME	E_NAME
John Smith	Alumni Meet
Sara Khan	Career Fair
David Lee	Workshop

3 rows returned in 0.00 seconds

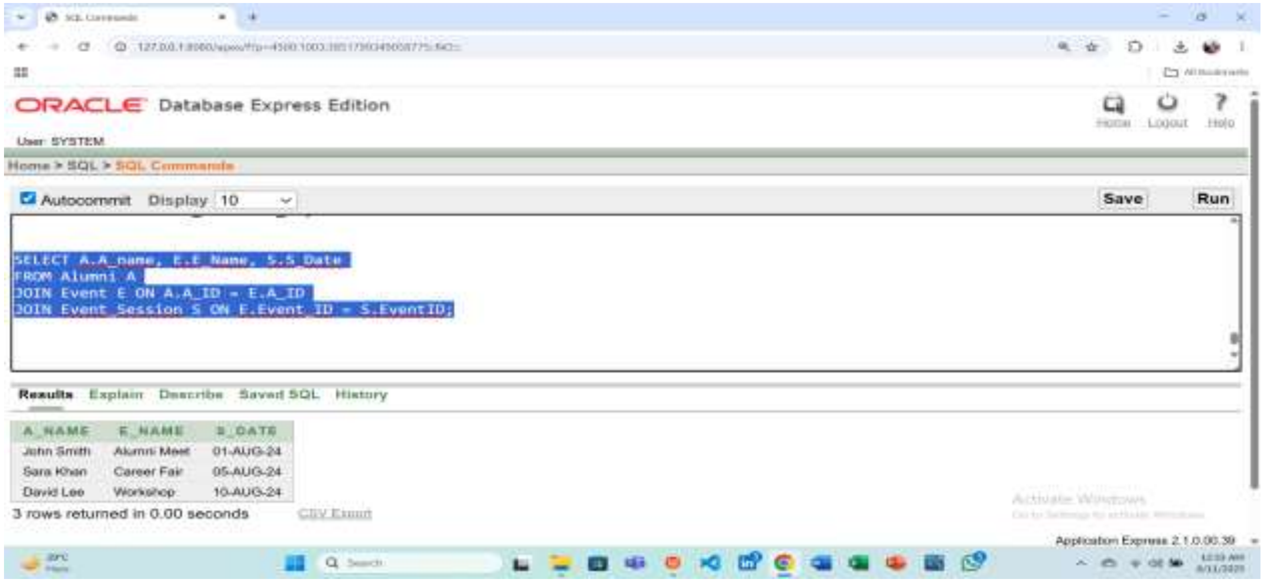
Question(2): Display alumni names, event names, and session dates for all scheduled sessions.
Answer:

SELECT A.A_name, E.E_Name, S.S_Date

FROM Alumni A

JOIN Event E ON A.A_ID = E.A_ID

JOIN Event_Session S ON E.Event_ID = S.EventID;


ORACLE Database Express Edition
User: SYSTEM
Home > SQL > SQL Commands
Autocommit Display 10
Save
Run
Results Explain Describe Saved SQL History
CSV Export

```
SELECT A.A_name, E.E_Name, S.S_Date
FROM Alumni A
JOIN Event E ON A.A_ID = E.A_ID
JOIN Event_Session S ON E.Event_ID = S.EventID;
```

A_NAME	E_NAME	S_DATE
John Smith	Alumni Meet	01-AUG-24
Sara Khan	Career Fair	05-AUG-24
David Lee	Workshop	10-AUG-24

3 rows returned in 0.00 seconds

5. **Query Writing (continuation of Question 4) (Write down the answer only. Give full screenshot of the Oracle 10g Homepage that contains the answer and result)**

   -All screenshots MUST include the DATE and TIME feature from the screen of the machine (PC, Laptop etc.) used

   PL/SQL

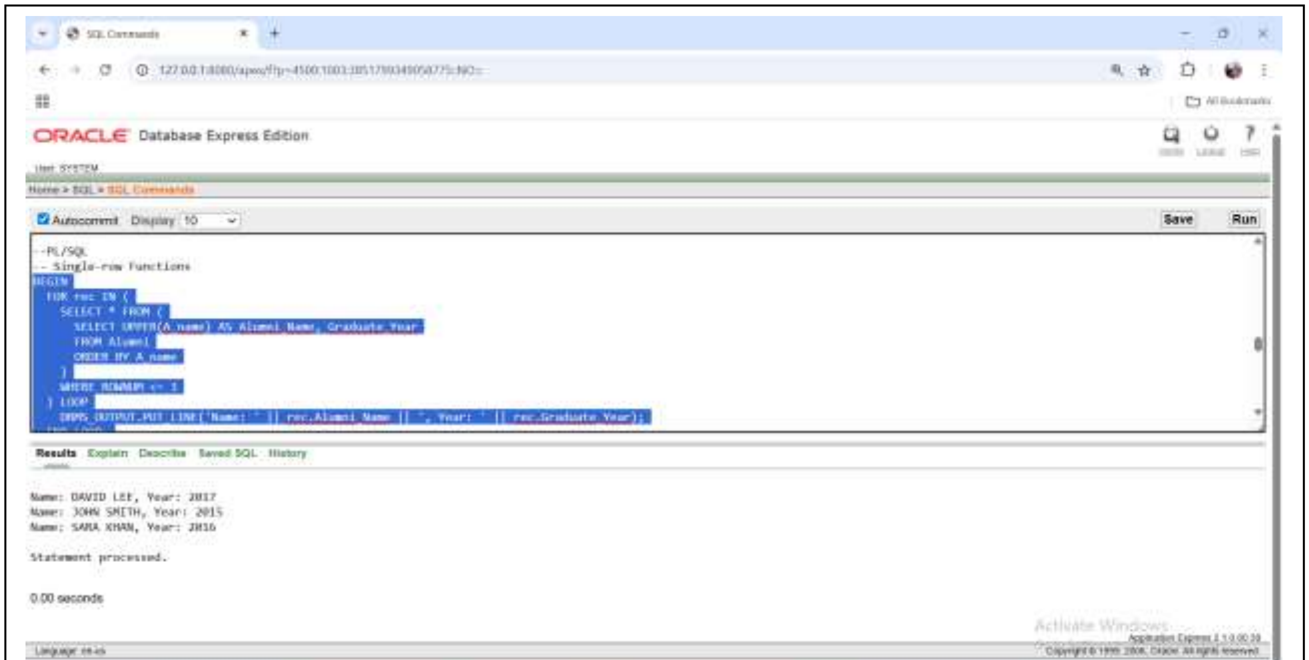
   1. **Convert the SQLs of Question 4 into equivalent PL/SQL code**
   2. **For this part, 8 PL/SQL code must be submitted**

Answer Box 5:

1. Single-Row Functions

Question(1): Display the names of alumni in uppercase and show their graduation year.
Answer:

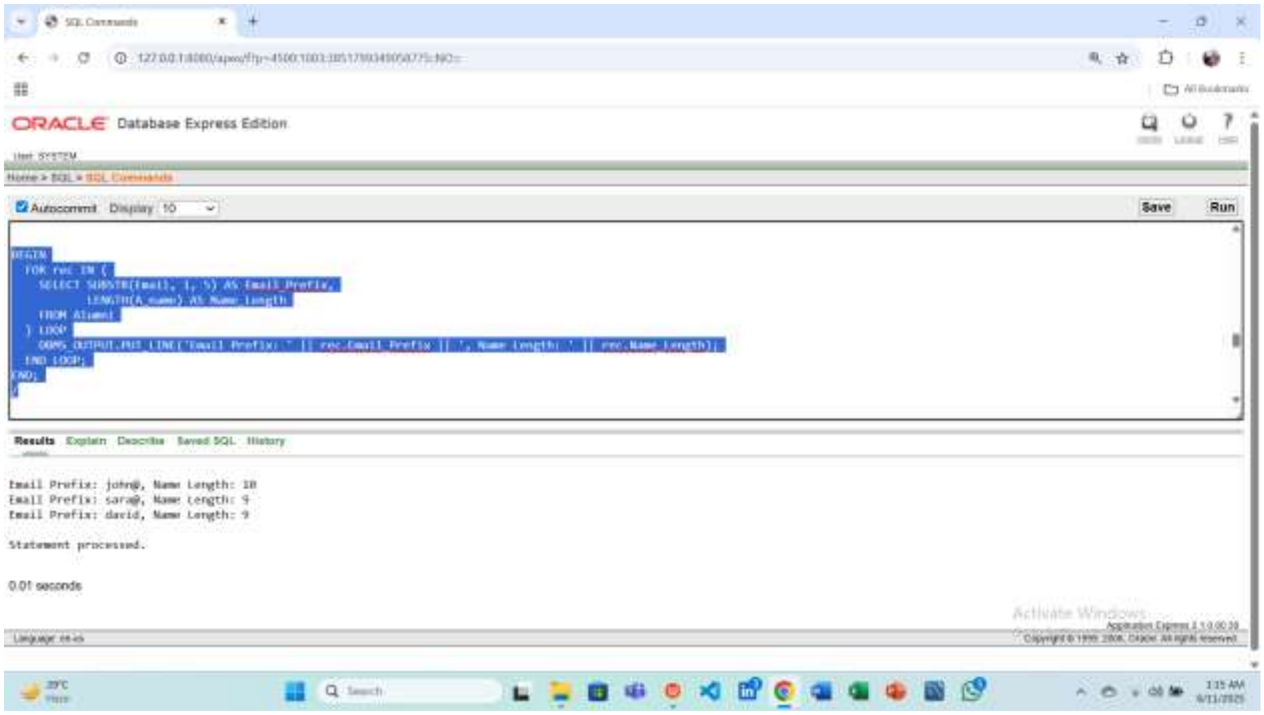
```
BEGIN

 FOR rec IN (

  SELECT * FROM (

   SELECT UPPER(A_name) AS Alumni_Name, Graduate_Year

   FROM Alumni

   ORDER BY A_name

  )

  WHERE ROWNUM <= 3

 ) LOOP

  DBMS_OUTPUT.PUT_LINE('Name: ' || rec.Alumni_Name || ', Year: ' || rec.Graduate_Year);

 END LOOP;

END;

/
```


ORACLE Database Express Edition
User: SYSTEM
Home > SQL > SQL Commands
Autocommit Display 10
Save
Run
Results Explain Describe Saved SQL History

```
--PL/SQL
-- Single-row Functions
BEGIN
  FOR rec IN (
    SELECT * FROM (
      SELECT UPPER(A_name) AS Alumni_Name, Graduate_Year
      FROM Alumni
      ORDER BY A_name
    )
    WHERE ROWNUM <= 3
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Name: ' || rec.Alumni_Name || ', Year: ' || rec.Graduate_Year);
```

```
Name: DAVID LEE, Year: 2017
Name: JOHN SMITH, Year: 2015
Name: SARA KHAN, Year: 2016

Statement processed.
```

0.00 seconds



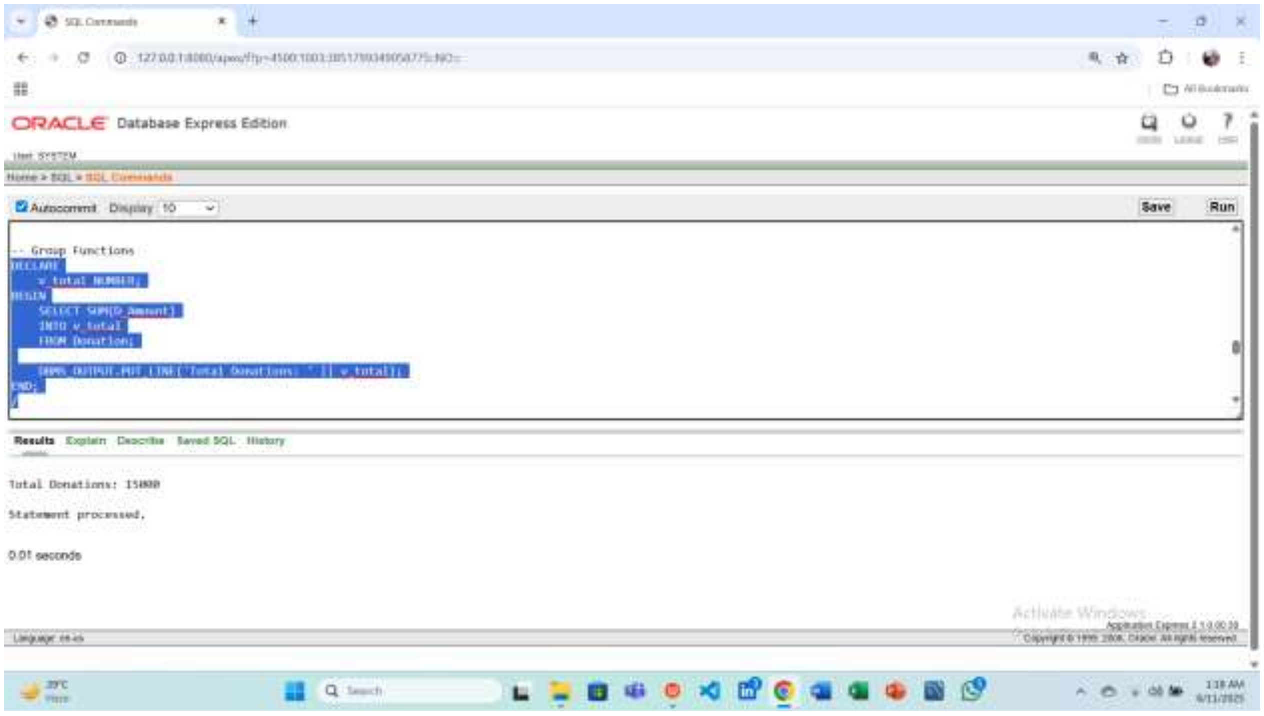
```
BEGIN
  FOR rec IN (
    SELECT SUBSTR(Email, 1, 5) AS Email_Prefix,
           LENGTH(A_name) AS Name_Length
    FROM Alumni
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Email Prefix: ' || rec.Email_Prefix || ', Name Length: ' || rec.Name_Length);
  END LOOP;
END;
/
```

```
Email Prefix: john@, Name Length: 10
Email Prefix: sara@, Name Length: 9
Email Prefix: david, Name Length: 9

Statement processed.

0.01 seconds
```

2. Group Functions

Question(1): Find the total donation amount received from all alumni.
Answer:

```
DECLARE

   v_total NUMBER;

BEGIN

   SELECT SUM(D_Amount)

   INTO v_total

   FROM Donation;



   DBMS_OUTPUT.PUT_LINE('Total Donations: ' || v_total);

END;

/
```

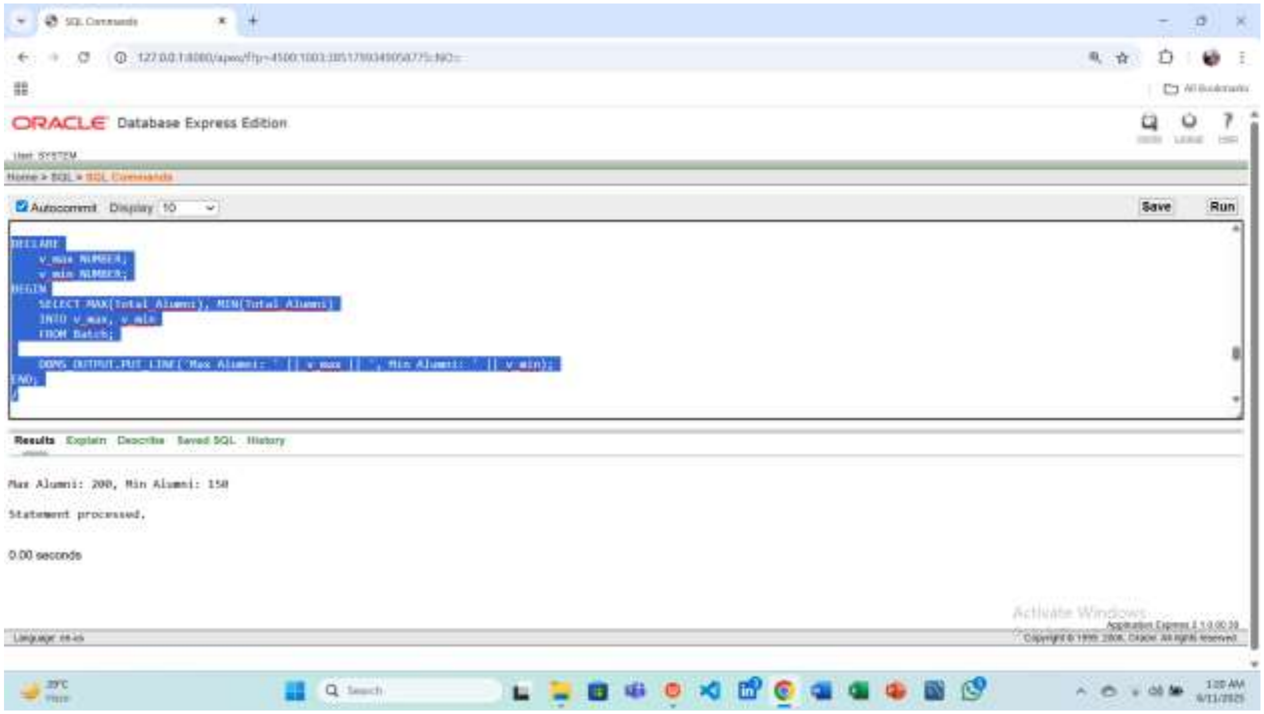


```
-- Group Functions
DECLARE
    v_total NUMBER;
BEGIN
    SELECT SUM(D_Amount)
    INTO v_total
    FROM Donation;

    DBMS_OUTPUT.PUT_LINE('Total Donations: ' || v_total);
END;
/
```

```
Total Donations: 15000

Statement processed.

0.01 seconds
```

Question(2): Find the maximum and minimum total alumni count among all batches.
Answer:

```
DECLARE

   v_max NUMBER;

   v_min NUMBER;

BEGIN

   SELECT MAX(Total_Alumni), MIN(Total_Alumni)

   INTO v_max, v_min

   FROM Batch;



   DBMS_OUTPUT.PUT_LINE('Max Alumni: ' || v_max || ', Min Alumni: ' || v_min);

END;

/
```



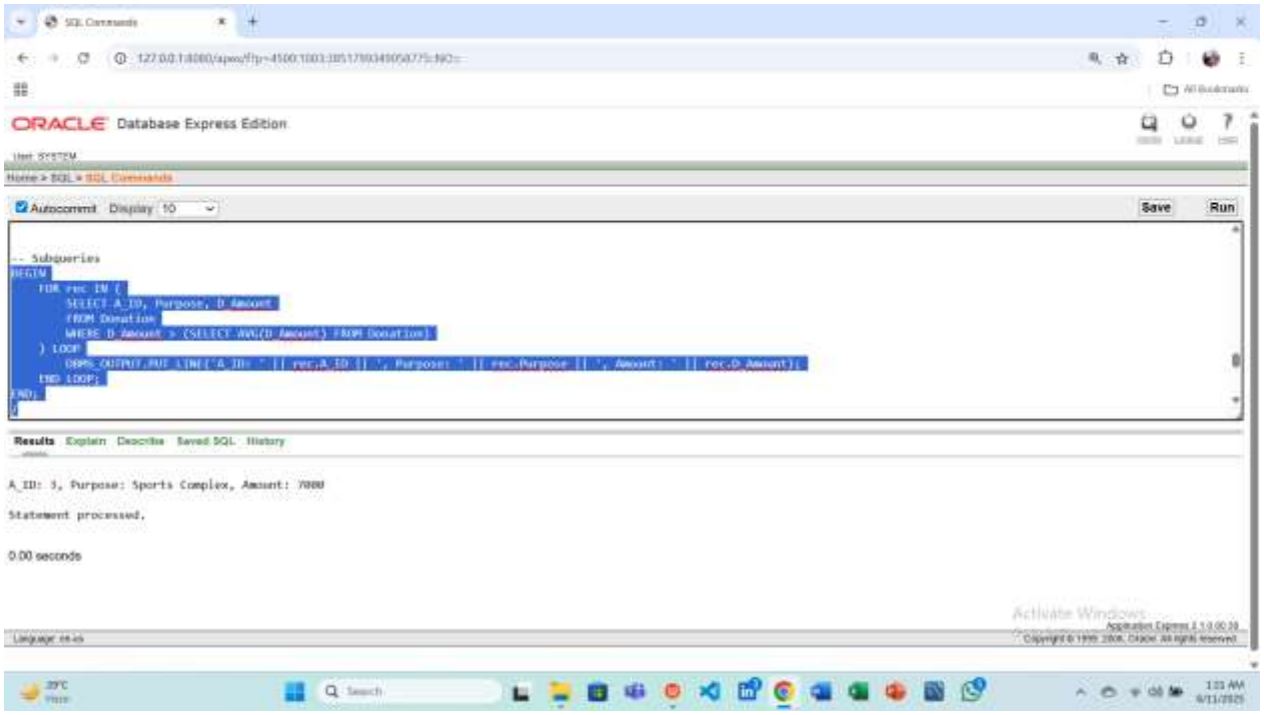
```
DECLARE
    v_max NUMBER;
    v_min NUMBER;
BEGIN
    SELECT MAX(Total_Alumni), MIN(Total_Alumni)
    INTO v_max, v_min
    FROM Batch;

    DBMS_OUTPUT.PUT_LINE('Max Alumni: ' || v_max || ', Min Alumni: ' || v_min);
END;
/
```

```
Max Alumni: 200, Min Alumni: 150

Statement processed.

0.00 seconds
```

3. Subqueries

Question(1): Display the alumni who donated more than the average donation amount.
Answer:

```
BEGIN

   FOR rec IN (

      SELECT A_ID, Purpose, D_Amount

      FROM Donation

      WHERE D_Amount > (SELECT AVG(D_Amount) FROM Donation)

   ) LOOP

      DBMS_OUTPUT.PUT_LINE('A_ID: ' || rec.A_ID || ', Purpose: ' || rec.Purpose || ', Amount: ' ||
rec.D_Amount);

   END LOOP;

END;

/
```

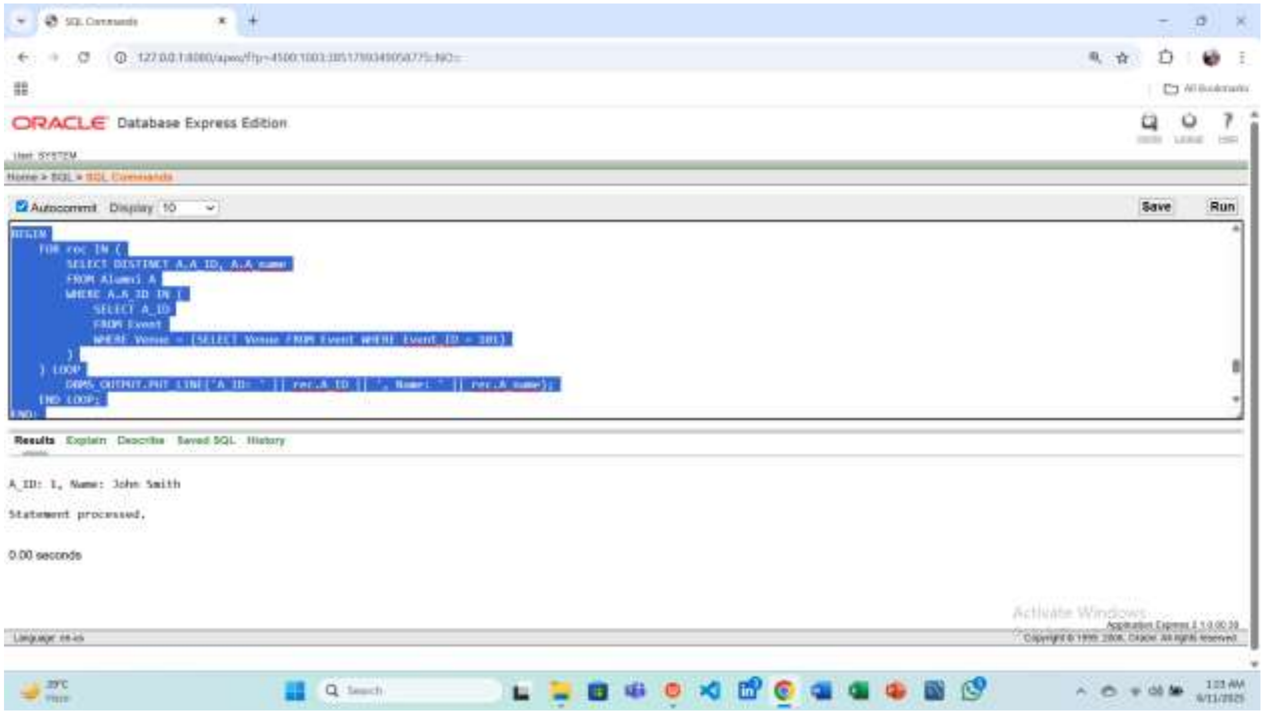

ORACLE Database Express Edition
Home > SQL > SQL Commands
Autocommit Display 10
Save
Run
Results Explain Describe Saved SQL History

```
-- Subqueries
BEGIN
    FOR rec IN (
        SELECT A_ID, Purpose, D_Amount
        FROM Donation
        WHERE D_Amount > (SELECT AVG(D_Amount) FROM Donation)
    ) LOOP
        DBMS_OUTPUT.PUT_LINE('A_ID: ' || rec.A_ID || ', Purpose: ' || rec.Purpose || ', Amount: ' || rec.D_Amount);
    END LOOP;
END;
/
```

```
A_ID: 3, Purpose: Sports Complex, Amount: 7000

Statement processed.

0.00 seconds
```

Question(2): Display alumni who participated in events with the same venue as event ID 101.
Answer:

```
BEGIN

  FOR rec IN (

    SELECT DISTINCT A.A_ID, A.A_name

    FROM Alumni A

    WHERE A.A_ID IN (

      SELECT A_ID

      FROM Event

      WHERE Venue = (SELECT Venue FROM Event WHERE Event_ID = 101)

    )

  ) LOOP

    DBMS_OUTPUT.PUT_LINE('A_ID: ' || rec.A_ID || ', Name: ' || rec.A_name);

  END LOOP;

END;

/
```



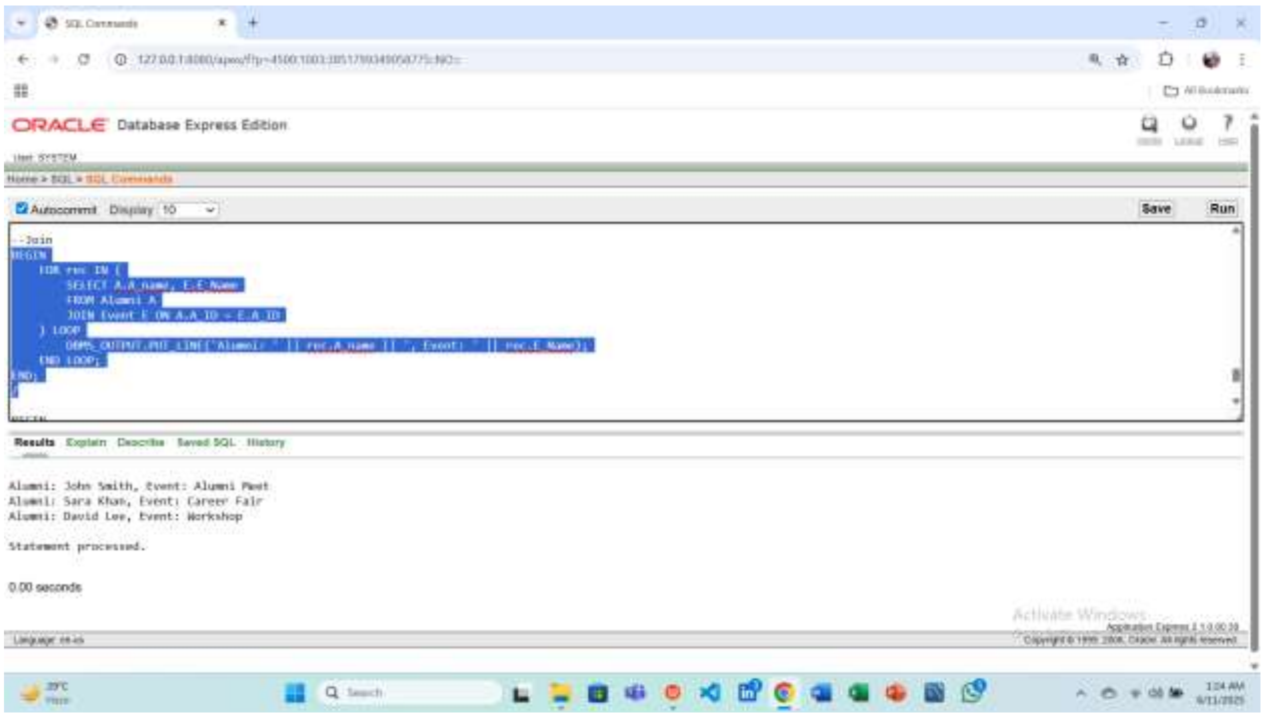
```
BEGIN
    FOR rec IN (
        SELECT DISTINCT A.A_ID, A.A_name
        FROM Alumni A
        WHERE A.A_ID IN (
            SELECT A_ID
            FROM Event
            WHERE Venue = (SELECT Venue FROM Event WHERE Event_ID = 101)
        )
    ) LOOP
        DBMS_OUTPUT.PUT_LINE('A_ID: ' || rec.A_ID || ', Name: ' || rec.A_name);
    END LOOP;
END;
```

```
A_ID: 1, Name: John Smith

Statement processed.

0.00 seconds
```

4. Joins

Question(1): Display all alumni names along with the event names they are associated with.
Answer:

```
BEGIN

   FOR rec IN (

      SELECT A.A_name, E.E_Name

      FROM Alumni A

      JOIN Event E ON A.A_ID = E.A_ID

   ) LOOP

      DBMS_OUTPUT.PUT_LINE('Alumni: ' || rec.A_name || ', Event: ' || rec.E_Name);

   END LOOP;

END;

/
```

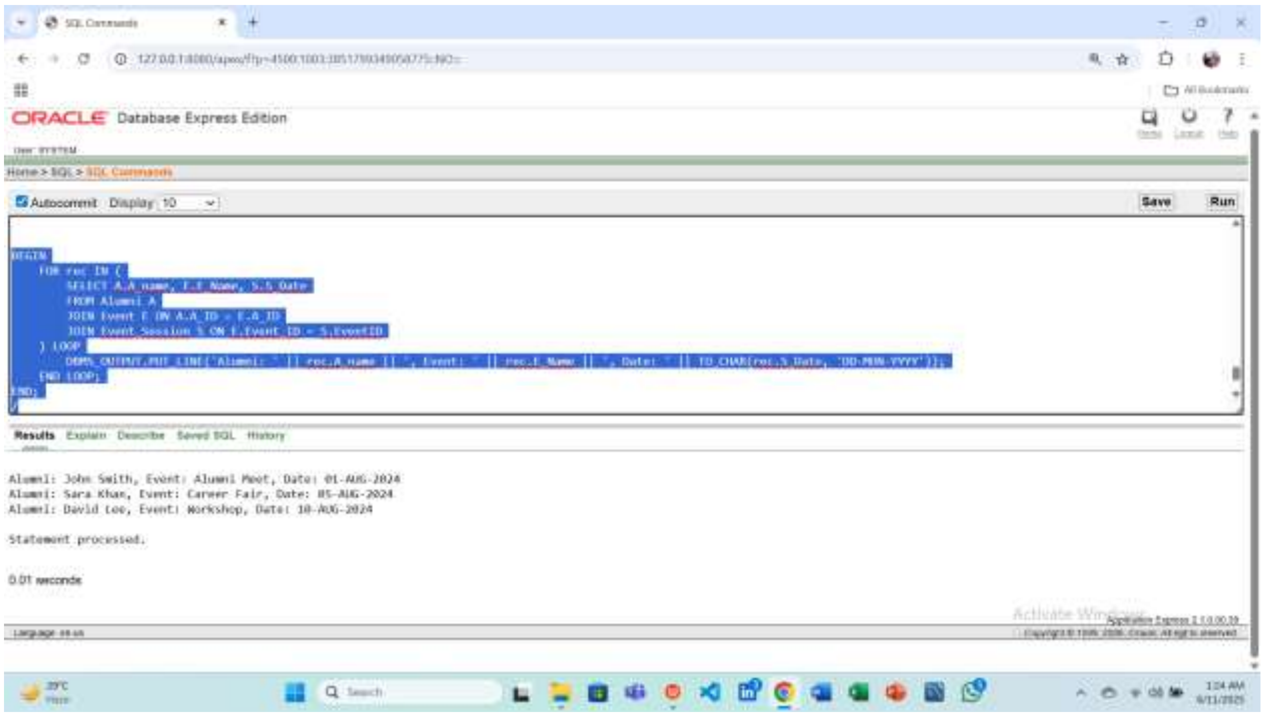


```
--Join
BEGIN
    FOR rec IN (
        SELECT A.A_name, E.E_Name
        FROM Alumni A
        JOIN Event E ON A.A_ID = E.A_ID
    ) LOOP
        DBMS_OUTPUT.PUT_LINE('Alumni: ' || rec.A_name || ', Event: ' || rec.E_Name);
    END LOOP;
END;
/
```

```
Alumni: John Smith, Event: Alumni Meet
Alumni: Sara Khan, Event: Career Fair
Alumni: David Lee, Event: Workshop

Statement processed.

0.00 seconds
```

Question(2): Display alumni names, event names, and session dates for all scheduled sessions.
Answer:

```
BEGIN
   FOR rec IN (
       SELECT A.A_name, E.E_Name, S.S_Date
       FROM Alumni A
       JOIN Event E ON A.A_ID = E.A_ID
       JOIN Event_Session S ON E.Event_ID = S.EventID
   ) LOOP
       DBMS_OUTPUT.PUT_LINE('Alumni: ' || rec.A_name || ', Event: ' || rec.E_Name || ', Date: ' ||
TO_CHAR(rec.S_Date, 'DD-MON-YYYY'));
   END LOOP;
END;
/
```



SQL Commands
ORACLE Database Express Edition
User: SYSTEM
Home > SQL > SQL Commands
Autocommit Display 10
Save
Run
Results Explain Describe Saved SQL History

```
BEGIN
    FOR rec IN (
        SELECT A.A_name, E.E_Name, S.S_Date
        FROM Alumni A
        JOIN Event E ON A.A_ID = E.A_ID
        JOIN Event_Session S ON E.Event_ID = S.EventID
    ) LOOP
        DBMS_OUTPUT.PUT_LINE('Alumni: ' || rec.A_name || ', Event: ' || rec.E_Name || ', Date: ' || TO_CHAR(rec.S_Date, 'DD-MON-YYYY'));
    END LOOP;
END;
/
```

```
Alumni: John Smith, Event: Alumni Meet, Date: 01-AUG-2024
Alumni: Sara Khan, Event: Career Fair, Date: 05-AUG-2024
Alumni: David Lee, Event: Workshop, Date: 10-AUG-2024

Statement processed.
```

0.01 seconds