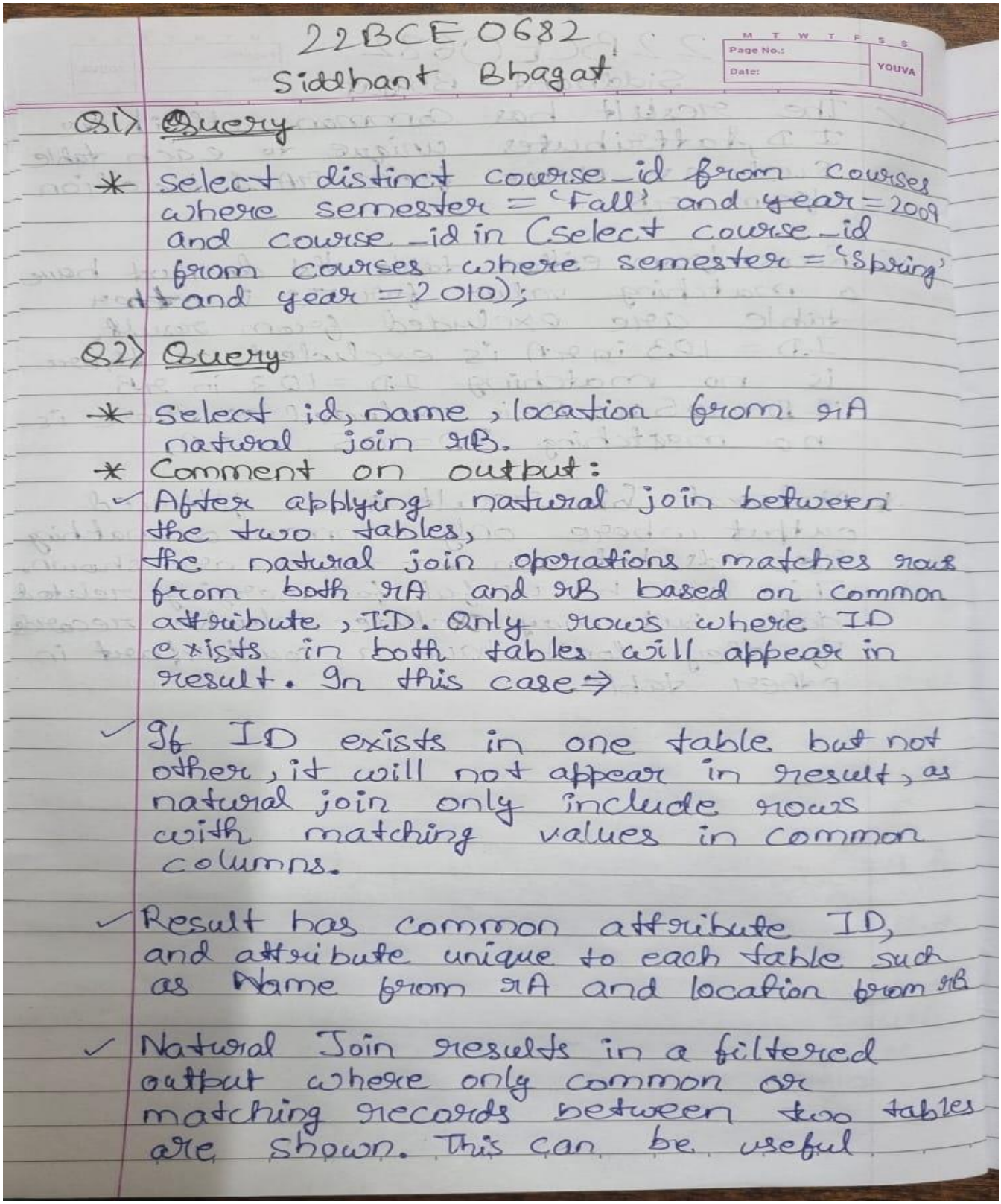
HANDWRITTEN SUBMISSION: (2 pages)



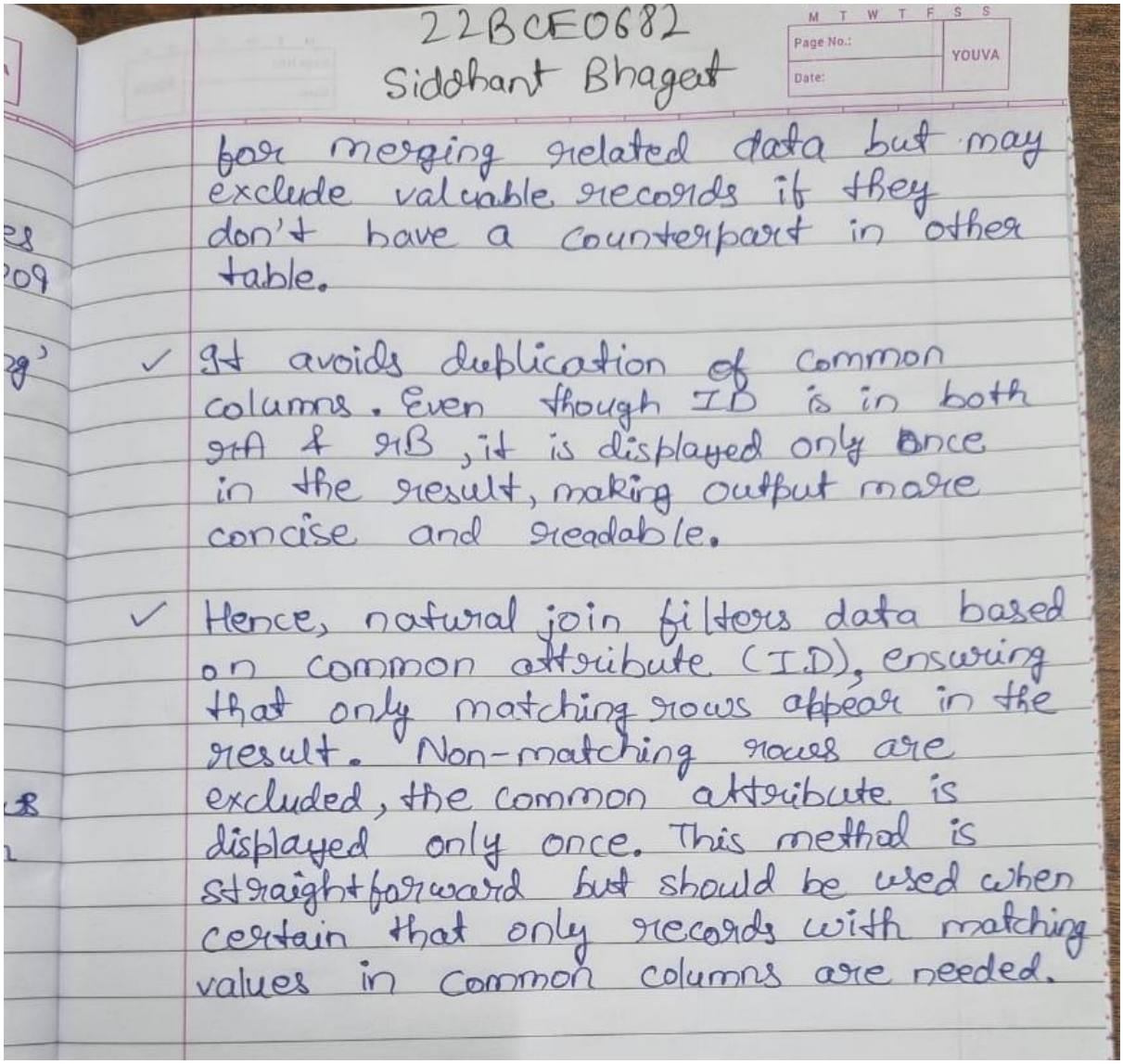
22BCE0682
Siddhant Bhagat

Q1) Query

* Select distinct course_id from courses where semester = 'Fall' and year = 2009 and course_id in (select course_id from courses where semester = 'Spring' and year = 2010);

Q2) Query

* Select id, name, location from rA natural join rB.
* Comment on output:
  - ✓ After applying natural join between the two tables, the natural join operations matches rows from both rA and rB based on common attribute, ID. Only rows where ID exists in both tables will appear in result. In this case ⇒
  - ✓ If ID exists in one table but not other, it will not appear in result, as natural join only include rows with matching values in common columns.
  - ✓ Result has common attribute ID, and attribute unique to each table such as Name from rA and location from rB
  - ✓ Natural Join results in a filtered output where only common or matching records between two tables are shown. This can be useful


22BCE0682
Siddhant Bhagat

for merging related data but may exclude valuable records if they don't have a counterpart in other table.

✓ It avoids duplication of common columns. Even though ID is in both rA & rB, it is displayed only once in the result, making output more concise and readable.

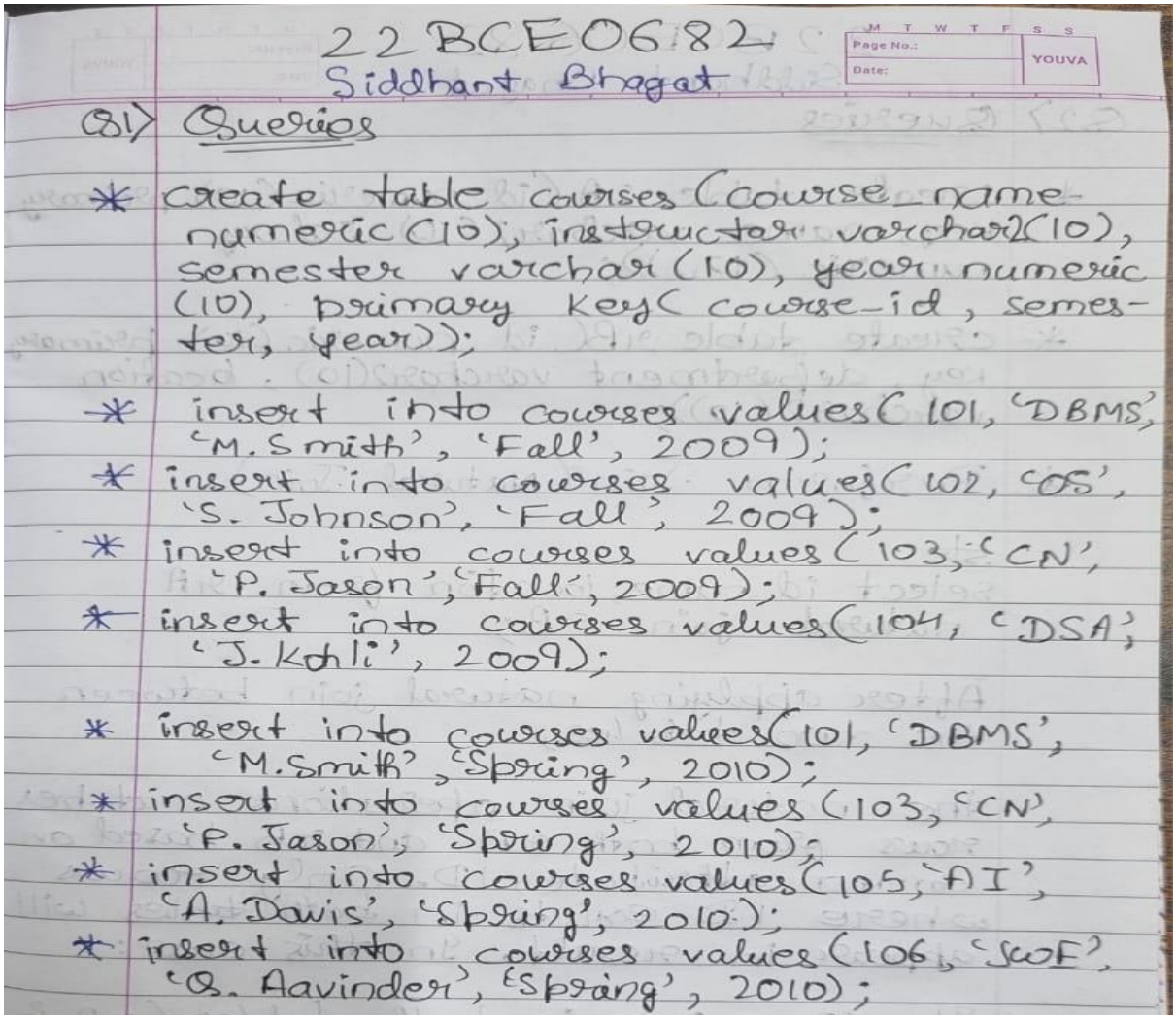
✓ Hence, natural join filters data based on common attribute (ID), ensuring that only matching rows appear in the result. Non-matching rows are excluded, the common attribute is displayed only once. This method is straightforward but should be used when certain that only records with matching values in common columns are needed.

TOTAL BREAKDOWN OF QUESTIONS WITH SYSTEM SCREENSHOTS

Q1)

Write SQL query for the following:

- Find all the courses taught in the both the Fall 2009 and Spring 2010 semesters **(5m).**



22BCE0682
Siddhant Bhagat

Q1) Queries

* create table courses (course_name numeric(10), instructor varchar2(10), semester varchar(10), year numeric (10), primary key(course-id , semester, year));
* insert into courses values(101, 'DBMS', 'M.Smith', 'Fall', 2009);
* insert into courses values(102, 'OS', 'S. Johnson', 'Fall', 2009);
* insert into courses values(103, 'CN', 'P. Jason', 'Fall', 2009);
* insert into courses values(104, 'DSA', 'J. Kohli', 2009);
* insert into courses values(101, 'DBMS', 'M.Smith', 'Spring', 2010);
* insert into courses values (103, 'CN', 'P. Jason', 'Spring', 2010);
* insert into courses values(105, 'AI', 'A. Davis', 'Spring', 2010);
* insert into courses values (106, 'SWE', 'S. Aavinder', 'Spring', 2010);

TABLE CREATED

```
SQL> create table courses(course_id numeric(10), course_name varchar2(10), instructor varchar2(10), semester varchar(10), year numeric(10), primary key(co
urse_id, semester,year));

Table created.
```

INSERTING VALUES

```
SQL> insert into courses values(101,'DBMS','M.Smith','Fall',2009);

1 row created.

SQL> insert into courses values(102,'OS','S.Johnson','Fall',2009);

1 row created.

SQL> insert into courses values(103,'CN','P.Jason','Fall',2009);

1 row created.

SQL> insert into courses values(104,'DSA','J.Kohli','Fall',2009);

1 row created.

SQL> insert into courses values(101,'DBMS','M.Smith','Spring',2010);

1 row created.
```

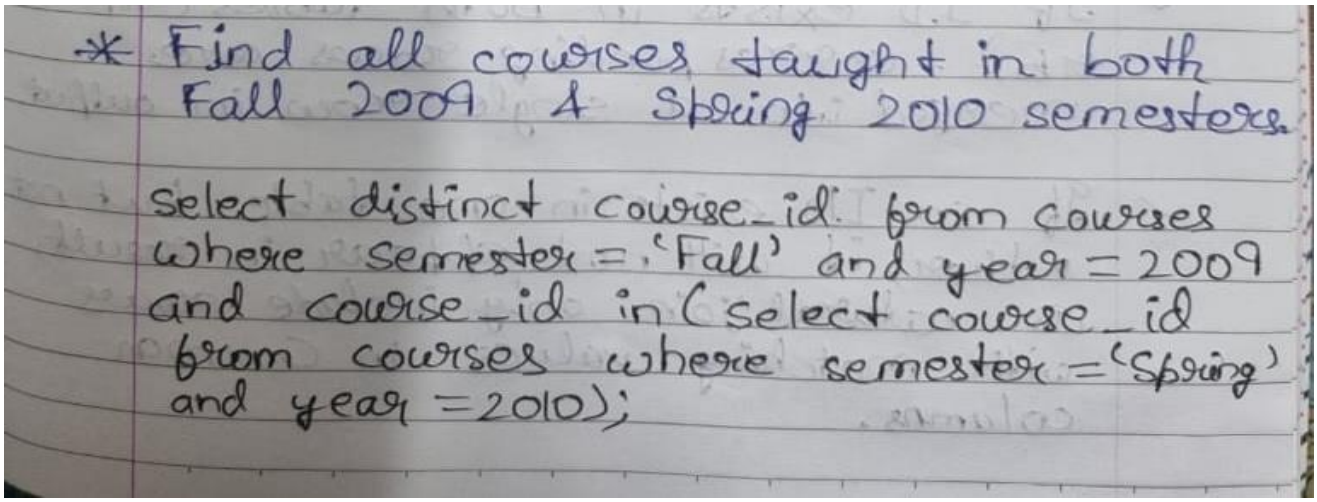
```
SQL> insert into courses values(103,'CN','P.Jason','Spring',2010);

1 row created.

SQL> insert into courses values(105,'AI','A.Davis','Spring',2010);

1 row created.

SQL> insert into courses values(106,'SWE','Q.Aavinder','Spring',2010);

1 row created.
```

DISPLAYING VALUES IN TABLE COURSES

```
 COURSE_ID COURSE_NAM INSTRUCTOR SEMESTER         YEAR
---------- ---------- ---------- ---------- ----------
       101 DBMS       M.Smith    Fall             2009
       102 OS         S.Johnson  Fall             2009
       103 CN         P.Jason    Fall             2009
       104 DSA        J.Kohli    Fall             2009
       101 DBMS       M.Smith    Spring           2010
       103 CN         P.Jason    Spring           2010
       105 AI         A.Davis    Spring           2010
       106 SWE        Q.Aavinder Spring           2010

8 rows selected.
```

Finding all the courses taught in the both the Fall 2009 and Spring 2010 semesters

```
SQL> select distinct course_id from courses where semester='Fall' and year=2009 and course_id in(select course_id from courses where semester='Spring'
 and year =2010);

 COURSE_ID
---------
      101
      103
```



Q2)

- Assume any two relation states with 3 attributes each - Apply natural join and display only the common attribute and one more attribute from each relation – output should have just 3 attributes. Comment on the output. **(5m)**

CREATION AND DISPLAYING RELATION STATE rA

```
SQL> create table rA(id numeric(10) primary key, name varchar2(10), salary
numeric(10,2));

Table created.
```

```
SQL> insert into rA values(101,'Bob',50000);

1 row created.

SQL> insert into rA values(102,'Jasion',40000);

1 row created.

SQL> insert into rA values(103,'Mason',10000);

1 row created.

SQL> insert into rA values(104,'Mount',80000);

1 row created.
```

DISPLAYING THE TABLE

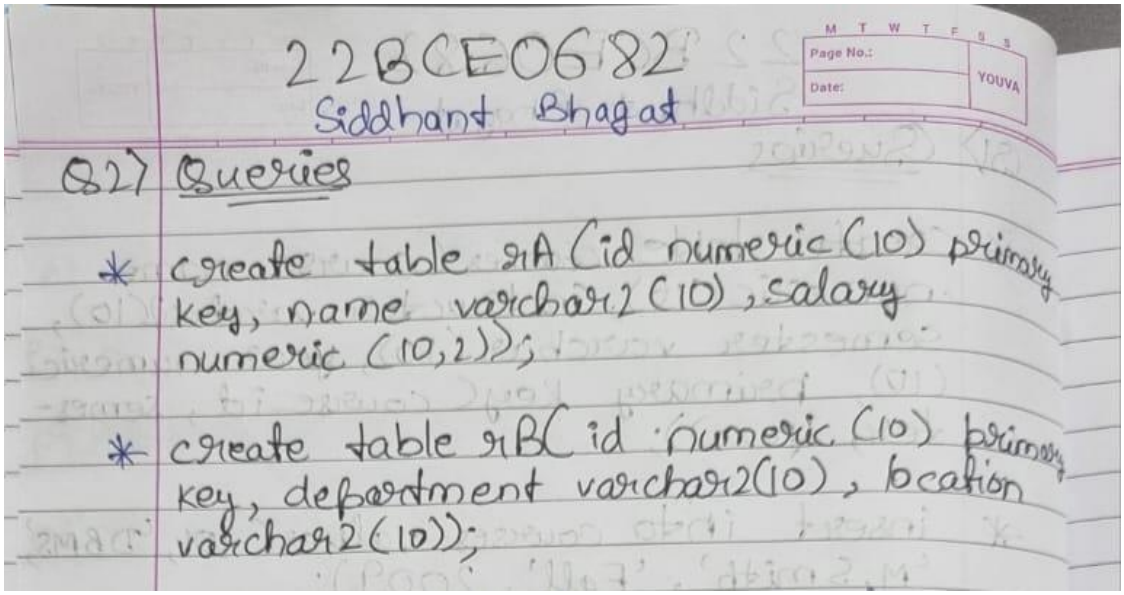
```
SQL> select * from rA;

        ID NAME            SALARY
---------- ---------- ----------
       101 Bob             50000
       102 Jasion          40000
       103 Mason           10000
       104 Mount           80000
```

SIDDHANT BHAGAT 22BCE0682

CREATION AND DISPLAYING RELATION STATE rB

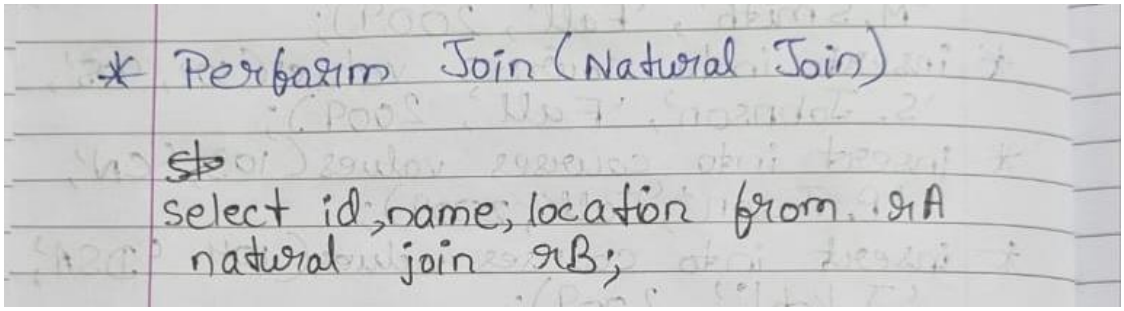
```
SQL> create table rB(id numeric(10) primary key, department varchar2(10), location varchar2(10));

Table created.

SQL> insert into rB values(101,'HR','New York');

1 row created.

SQL> insert into rB values(102,'IT','Dehli');

1 row created.

SQL> insert into rB values(105,'Finance','Atalanta');

1 row created.

SQL> insert into rB values(104,'Sales','Mumbai');

1 row created.
```

```
SQL> select * from rB;

        ID DEPARTMENT LOCATION
---------- ---------- ----------
       101 HR         New York
       102 IT         Dehli
       105 Finance    Atalanta
       104 Sales      Mumbai
```



PERFORMING NATURAL JOIN ON RA AND RB

```
SQL> select id,name,location from rA natural join rB;
```




```
        ID NAME       LOCATION
---------- ---------- ----------
       101 Bob        New York
       102 Jasion     Dehli
       104 Mount      Mumbai
```

COMMENTS:(explain)

* Comment on output:

✓ After applying natural join between the two tables, the natural join operations matches rows from both rA and rB based on common attribute, ID. Only rows where ID exists in both tables will appear in result. In this case ⇒

✓ If ID exists in one table but not other, it will not appear in result, as natural join only include rows with matching values in common columns.

✓ Result has common attribute ID, and attribute unique to each table such as Name from rA and location from rB

✓ Natural Join results in a filtered output where only common or matching records between two tables are shown. This can be useful

22BCE0682
Siddhant Bhagat

for merging related data but may exclude valuable records if they don't have a counterpart in other table.

✓ It avoids duplication of common columns. Even though ID is in both rA & rB, it is displayed only once in the result, making output more concise and readable.

✓ Hence, natural join filters data based on common attribute (I.D), ensuring that only matching rows appear in the result. Non-matching rows are excluded, the common attribute is displayed only once. This method is straightforward but should be used when certain that only records with matching values in common columns are needed.