

## ASSIGNMENT - III

### TITLE :

Design at least 10 SQL queries for suitable database application using SQL DML Statements.

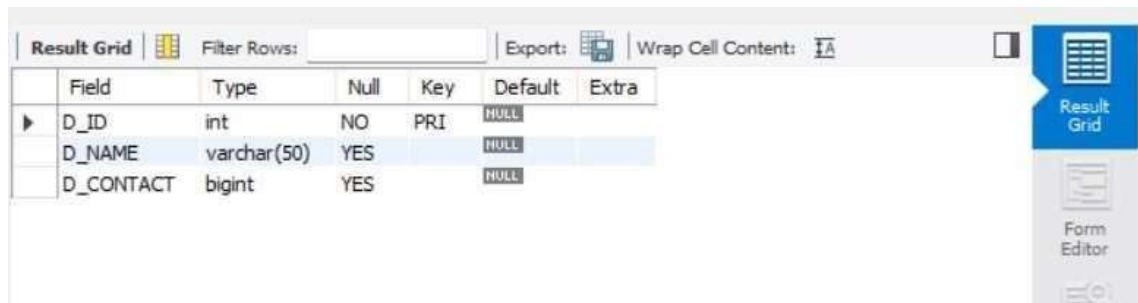
NAME : Shinde Shubham Dnyandev,      DIV : SY-B,      ROLL NO. : 23107121.

### QUERIES :

#### 1) CREATE :

a) CREATE DATABASE PR3;  
USE PR3;

```
CREATE TABLE DEPARTMENT(  
D_ID INT PRIMARY KEY,  
D_NAME VARCHAR(50),  
D_CONTACT BIGINT  
);  
DESC DEPARTMENT;
```



The screenshot shows a database management tool interface. At the top, there is a toolbar with options like 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. Below this is a table structure view for the 'DEPARTMENT' table. The table has three columns: 'D\_ID' (int, primary key, not null), 'D\_NAME' (varchar(50), yes null), and 'D\_CONTACT' (bigint, yes null). The 'Result Grid' button is visible on the right side of the interface.

Field	Type	Null	Key	Default	Extra
D_ID	int	NO	PRI		
D_NAME	varchar(50)	YES			
D_CONTACT	bigint	YES			

b) CREATE TABLE STUDENT(  
S\_ID INT PRIMARY KEY,  
NAME VARCHAR(50),  
ROLL\_NO INT,  
AGE INT,  
D\_ID INT,  
FOREIGN KEY (D\_ID) REFERENCES DEPARTMENT(D\_ID)  
);  
DESC STUDENT;

Field	Type	Null	Key	Default	Extra
S_ID	int	NO	PRI	<input type="text" value="NULL"/>	
NAME	varchar(50)	YES		<input type="text" value="NULL"/>	
ROLL_NO	int	YES		<input type="text" value="NULL"/>	
AGE	int	YES		<input type="text" value="NULL"/>	
D_ID	int	YES	MUL	<input type="text" value="NULL"/>	

## 2) INSERT :

### a) INSERT INTO DEPARTMENT VALUES

(465, 'AI&DS', 9544646642),  
 (847, 'COMP', 7656452365),  
 (364, 'IT', 8766647845),  
 (763, 'E&TC', 9867765894),  
 (273, 'MECH', 7768956543);

SELECT \* FROM DEPARTMENT;

D_ID	D_NAME	D_CONTACT
273	MECH	7768956543
364	IT	8766647845
465	AI&DS	9544646642
763	E&TC	9867765894
847	COMP	7656452365
<input type="text" value="NULL"/>	<input type="text" value="NULL"/>	<input type="text" value="NULL"/>

### b) INSERT INTO STUDENT VALUES

(6587, 'Ayush', 45, 24, 465),  
 (4938, 'Vinit', 07, 19, 847),  
 (3498, 'Aditya', 56, 27, 364),  
 (9476, 'Vivek', 19, 28, 763),  
 (3947, 'Yash', 60, 31, 273);

SELECT \* FROM STUDENT;

Result Grid					
Filter Rows:					
	S_ID	NAME	ROLL_NO	AGE	D_ID
▶	3498	Aditya	56	27	364
	3947	Yash	60	31	273
	4938	Vinit	7	34	847
	6587	Ayush	45	24	465
	9476	Vivek	19	28	763
•	NULL	NULL	NULL	NULL	NULL

### 3) UPDATE :

UPDATE STUDENT

SET AGE = AGE + 15

WHERE S\_ID = 4938;

SELECT \* FROM STUDENT;

Result Grid					
Filter Rows:					
	S_ID	NAME	ROLL_NO	AGE	D_ID
▶	3498	Aditya	56	27	364
	3947	Yash	60	31	273
	4938	Vinit	7	34	847
	6587	Ayush	45	24	465
	9476	Vivek	19	28	763
•	NULL	NULL	NULL	NULL	NULL

### 4) DELETE :

DELETE FROM STUDENT

WHERE S\_ID = 9476;

SELECT \* FROM STUDENT;

Result Grid					
Filter Rows:					
	S_ID	NAME	ROLL_NO	AGE	D_ID
▶	3498	Aditya	56	27	364
	3947	Yash	60	31	273
	4938	Vinit	7	34	847
	6587	Ayush	45	24	465
•	NULL	NULL	NULL	NULL	NULL

### 5) UNION :

```
SELECT NAME, D_ID  
FROM STUDENT  
WHERE D_ID = 364  
UNION  
SELECT NAME, D_ID  
FROM STUDENT  
WHERE D_ID = 273;
```



	NAME	D_ID
▶	Aditya	364
	Yash	273

### 6) SUM :

```
SELECT SUM(AGE) AS Total_Age FROM STUDENT;
```



	Total_Age
▶	116

### 7) GROUPBY :

```
SELECT S_ID, AVG(AGE) AS Avg_Age  
FROM STUDENT  
GROUP BY S_ID;
```



	S_ID	Avg_Age
▶	3498	27.0000
	3947	31.0000
	4938	34.0000
	6587	24.0000

### 8) MAX & MIN :

```
SELECT MAX(AGE) AS Highest_Age, MIN(AGE) AS Lowest_Age FROM STUDENT;
```



	Highest_Age	Lowest_Age
▶	34	24

### 9) COUNT :

```
SELECT D_ID, COUNT(*) AS Student_Count  
FROM STUDENT  
GROUP BY D_ID;
```

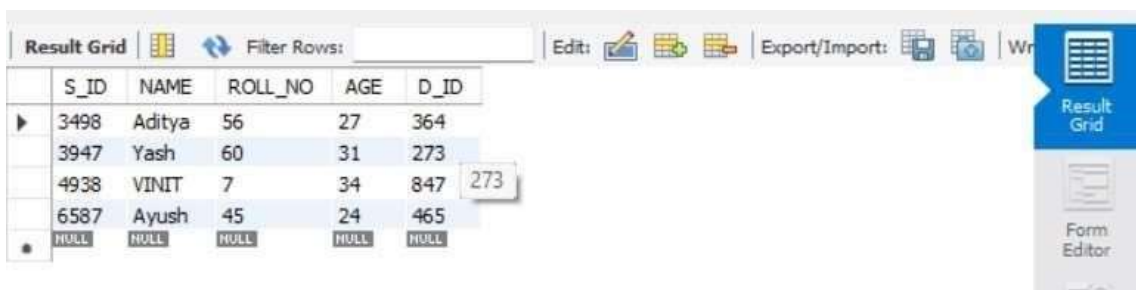


	D_ID	Student	Count
▶	273	1	
	364	1	
	465	1	
	847	1	

### 10) UPDATE (UPPERCASE) :

```
UPDATE STUDENT  
SET NAME = UPPER(NAME)  
WHERE S_ID = 4938;
```

```
SELECT * FROM STUDENT;
```

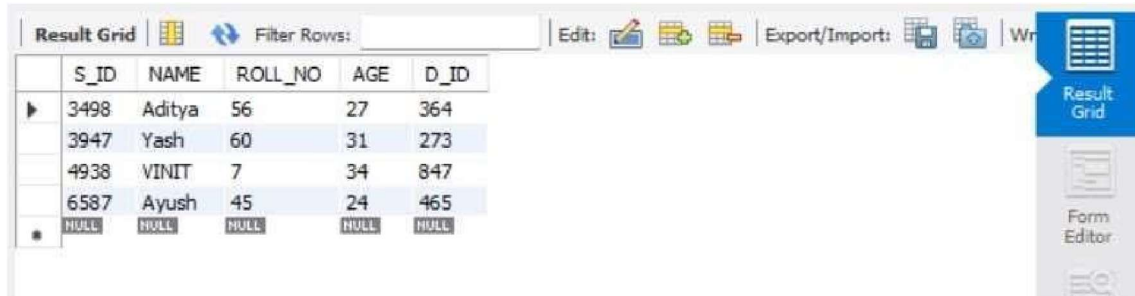


	S_ID	NAME	ROLL_NO	AGE	D_ID
▶	3498	Aditya	56	27	364
	3947	Yash	60	31	273
	4938	VINIT	7	34	847
	6587	Ayush	45	24	465
•	NULL	NULL	NULL	NULL	NULL

### 11) BITWISE OPERATER :

SELECT \* FROM STUDENT

WHERE NAME = 'Aditya' OR AGE > 20;



	S_ID	NAME	ROLL_NO	AGE	D_ID
▶	3498	Aditya	56	27	364
	3947	Yash	60	31	273
	4938	VINIT	7	34	847
	6587	Ayush	45	24	465
*	NULL	NULL	NULL	NULL	NULL

### 12) SPECIAL OPERATER :

a) SELECT \* FROM STUDENT

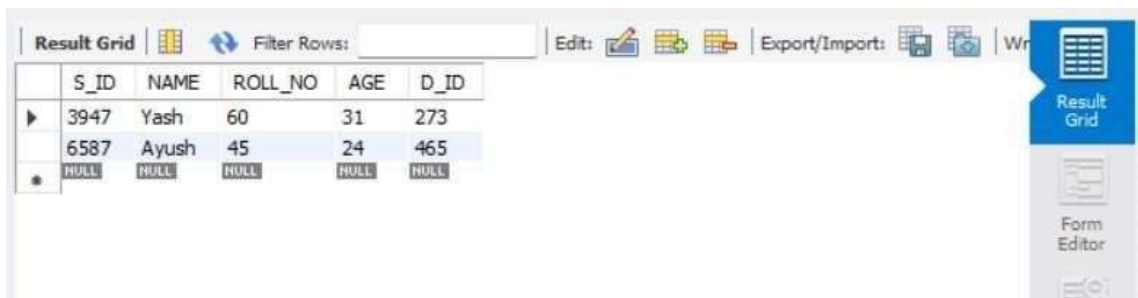
WHERE NAME LIKE 'V%';



	S_ID	NAME	ROLL_NO	AGE	D_ID
▶	4938	VINIT	7	34	847
*	NULL	NULL	NULL	NULL	NULL

b) SELECT \* FROM STUDENT

WHERE NAME LIKE '%sh';



	S_ID	NAME	ROLL_NO	AGE	D_ID
▶	3947	Yash	60	31	273
	6587	Ayush	45	24	465
*	NULL	NULL	NULL	NULL	NULL

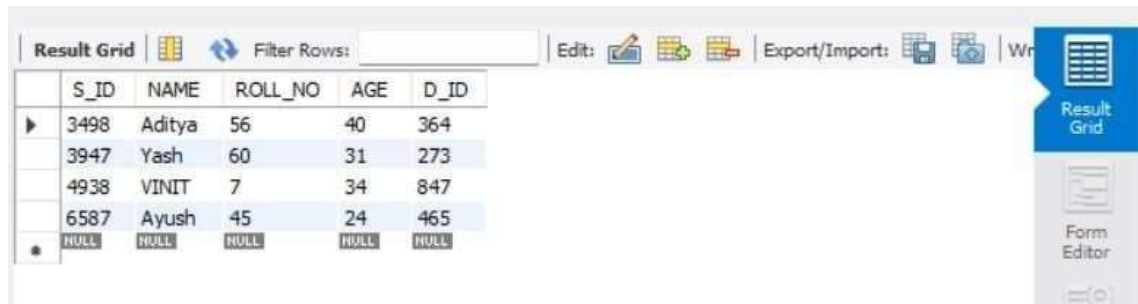
### 13) ROUND :

UPDATE STUDENT

SET AGE = ROUND(AGE / 20) \* 20

WHERE S\_ID = 3498;

SELECT \* FROM STUDENT;



The screenshot shows a database application interface with a 'Result Grid' tab. The grid displays the results of the SQL queries. The first four rows correspond to the data from the 'STUDENT' table after the update. The fifth row shows the result of the SELECT query, which is a row of NULL values.

S_ID	NAME	ROLL_NO	AGE	D_ID
3498	Aditya	56	40	364
3947	Yash	60	31	273
4938	VINIT	7	34	847
6587	Ayush	45	24	465
NULL	NULL	NULL	NULL	NULL

### 14) CONCAT :

SELECT CONCAT(NAME, ' ', AGE) AS NAME\_AND\_CONTACT

FROM STUDENT;



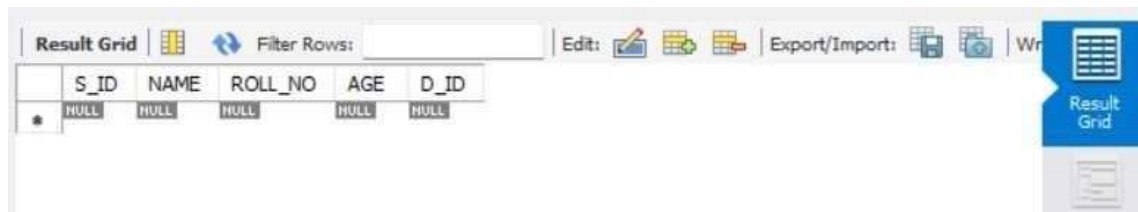
The screenshot shows a database application interface with a 'Result Grid' tab. The grid displays the results of the SQL query. The first four rows show the concatenated name and age for each student in the 'STUDENT' table.

NAME_AND_CONTACT
Aditya , 40
Yash , 31
VINIT , 34
Ayush , 24

### 15) IS NULL :

SELECT \* FROM STUDENT

WHERE ROLL\_NO IS NULL;



The screenshot shows a database application interface with a 'Result Grid' tab. The grid displays the results of the SQL query. The first row shows a row of NULL values, indicating that no records were found where ROLL\_NO is NULL.

S_ID	NAME	ROLL_NO	AGE	D_ID
NULL	NULL	NULL	NULL	NULL