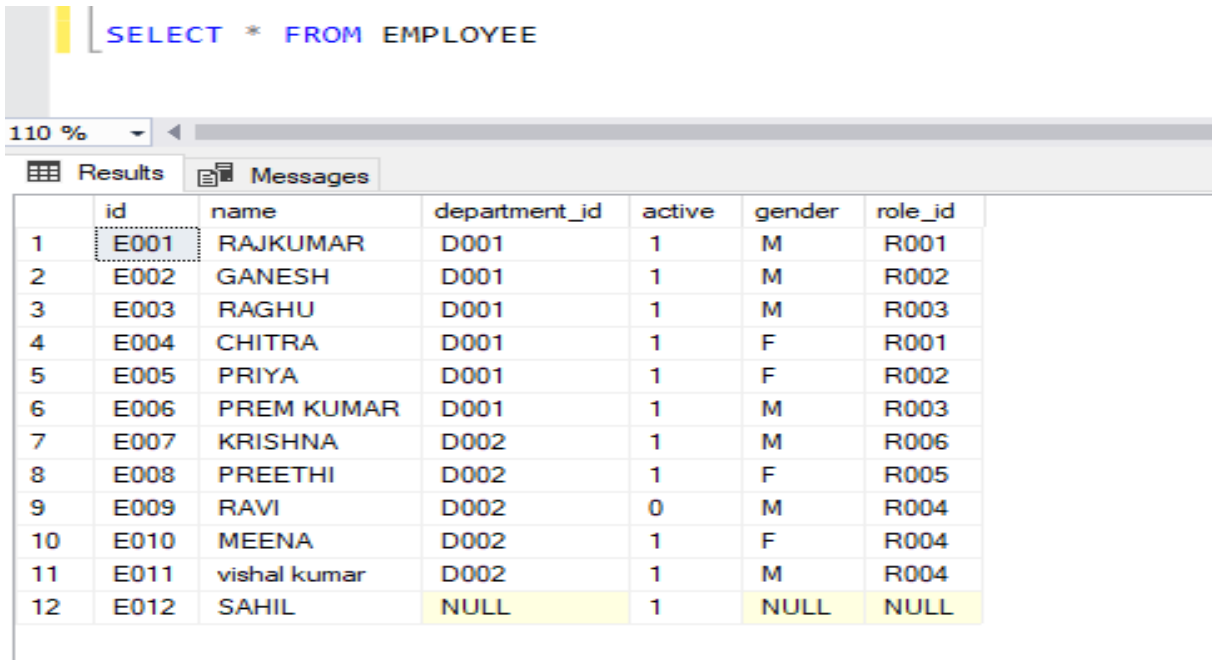


ASSIGNMENT

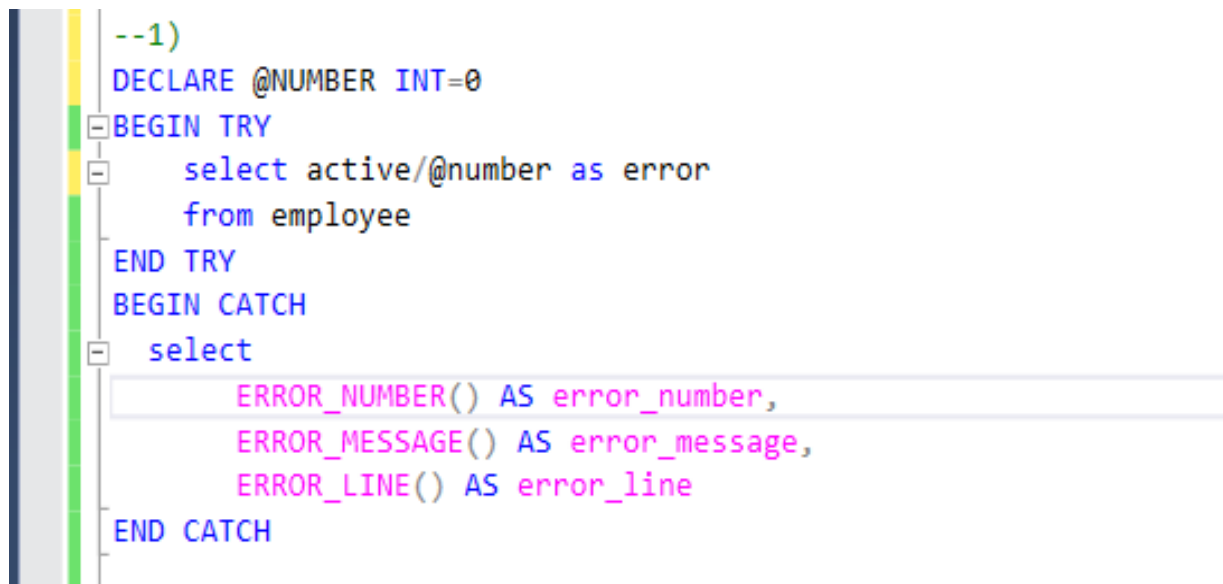
(ERROR HANDLING,PROCEDURE,TRANSACTION)



The screenshot shows a SQL query execution window with the query `SELECT * FROM EMPLOYEE`. The results are displayed in a table with 7 columns: id, name, department_id, active, gender, and role_id. The table contains 12 rows of data. The first row (id E001) is highlighted with a dashed border. The last row (id E012) has a yellow background.

	id	name	department_id	active	gender	role_id
1	E001	RAJKUMAR	D001	1	M	R001
2	E002	GANESH	D001	1	M	R002
3	E003	RAGHU	D001	1	M	R003
4	E004	CHITRA	D001	1	F	R001
5	E005	PRIYA	D001	1	F	R002
6	E006	PREM KUMAR	D001	1	M	R003
7	E007	KRISHNA	D002	1	M	R006
8	E008	PREETHI	D002	1	F	R005
9	E009	RAVI	D002	0	M	R004
10	E010	MEENA	D002	1	F	R004
11	E011	vishal kumar	D002	1	M	R004
12	E012	SAHIL	NULL	1	NULL	NULL

1. Write a SQL statement to implement error handling with the help of TRY and CATCH block to get the error details like error number, error message and error line.



The screenshot shows a SQL script editor with a TRY-CATCH block. The script declares a variable @NUMBER, attempts to execute a query that will cause a division by zero error, and then catches the error to retrieve its details.

```
--1)
DECLARE @NUMBER INT=0
BEGIN TRY
    select active/@number as error
    from employee
END TRY
BEGIN CATCH
    select
        ERROR_NUMBER() AS error_number,
        ERROR_MESSAGE() AS error_message,
        ERROR_LINE() AS error_line
END CATCH
```

Results Messages			
error			
	error_number	error_message	error_line
1	8134	Divide by zero error encountered.	4

2. Write a SQL statement to implement error handling in Stored Procedure and display the SP name.

```
--2)
CREATE PROCEDURE EMP_DETAILS @NUMBER INT
AS
BEGIN
    BEGIN TRY
        select id,name,active/@number
        from employee
    END TRY
    BEGIN CATCH
        select
            ERROR_NUMBER() AS error_number,
            ERROR_MESSAGE() AS error_message,
            ERROR_LINE() AS error_line,
            ERROR_PROCEDURE() AS procedure_name
    END CATCH
END
EXEC EMP_DETAILS 0
```

Results Messages				
	id	name	(No column name)	
	error_number	error_message	error_line	procedure_name
1	8134	Divide by zero error encountered.	5	EMP_DETAILS

3. Write a SQL statement to implement error handling with the help of TRY and CATCH inside TRANSACTION block to get the error details, error message and error line.

```
--3)
CREATE PROCEDURE show_emp_details @ID VARCHAR(10),@NAME VARCHAR(30),@ACTIVE INT
AS
BEGIN
    BEGIN TRANSACTION
    BEGIN TRY
        insert into employee(id,name,active) values (@ID,@NAME,@ACTIVE)
        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        IF (@@TRANCOUNT >0)
            ROLLBACK TRANSACTION

        SELECT
            ERROR_NUMBER() AS error_number,
            ERROR_MESSAGE() AS error_message,
            ERROR_LINE() AS error_line,
            ERROR_SEVERITY() AS error_severity,
            ERROR_STATE() AS error_state,
            ERROR_PROCEDURE() AS procedure_name
    END CATCH
END
```

	error_number	error_message	error_line	error_severity	error_state	procedure_name
1	2627	Violation of PRIMARY KEY constraint 'PK_employe...	6	14	1	show_emp_details

4. Write a Stored Procedure to implement user defined error handling.

```
--4)

CREATE PROCEDURE EMPDATA @ID VARCHAR(10),@NAME VARCHAR(30),@ACTIVE INT
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION
            insert into employee(id,name,active) values (@ID,@NAME,@ACTIVE)
        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        IF (@@TRANCOUNT >0)
            ROLLBACK TRANSACTION

        DECLARE @error_message VARCHAR(MAX)='Cannot insert duplicate values';

        THROW 50000, @error_message, 1;

    END CATCH
END

EXEC EMPDATA 'E012','SAHIL',1
```

Messages

(0 rows affected)
Msg 50000, Level 16, State 1, Procedure EMPDATA, Line 15 [Batch Start Line 73]
Cannot insert duplicate values

Completion time: 2024-02-27T12:20:13.1860039+05:30