

# **COURT CASES DATABASE**

## **CASE STUDY:**

Court Cases Database Management System

## **BACKGROUND:**

The judicial system involves a vast number of cases across various courts such as Civil, Criminal, Family, Commercial, Labor, and Tax. Traditionally, case information is recorded in registers and handled manually, which makes data management complex, error-prone, and time-consuming. Managing details such as case filing, judges, lawyers, hearings, parties involved, and related documents requires a systematic approach. Therefore, a Court Case Database Management System (DBMS) is proposed to modernize the process by implementing SQL-based database solutions for efficient case tracking, data storage, and retrieval.

## **PROJECT SCOPE:**

The scope of this project is to design and implement a Court Case Database Management System using SQL. The system will automate: Case registration and categorization (Civil, Criminal, Family, etc.), Allocation of cases to courts and judges, Tracking of hearings, outcomes, and proceedings, Management of documents, parties, and assigned lawyers, and Efficient queries for insights such as pending cases, judge allocation, and lawyer specialization. This project aims to enhance efficiency, accuracy, transparency, and accessibility in handling judicial records.

## **CHALLENGES:**

### **1. MANUAL DATA ENTRY & ERRORS:**

Handling numerous case records manually leads to duplication, data loss, and inconsistencies.

### **2. COMPLEX RELATIONSHIPS:**

Cases often involve multiple entities, including courts, judges, lawyers, hearings, and documents. Maintaining these interrelations is difficult without a database.

### 3. TRACKING HEARINGS & OUTCOMES:

Hearing outcomes, adjournments, and case progress tracking manually result in inefficiency and mismanagement.

### 4. DOCUMENT MANAGEMENT:

Storing and linking FIRs, petitions, evidence, and orders with respective cases becomes tedious without automation.

### 5. SECURITY & INTEGRITY:

Ensuring the authenticity of case details, preventing unauthorized access, and maintaining relational integrity are major concerns.

## **CONCLUSION:**

A Court Case Database Management System offers a structured way to manage judicial data, reducing manual workload and ensuring accuracy. It offers numerous benefits, including improved efficiency and transparency, accurate tracking of hearings and outcomes, simplified lawyer and judge assignment, secure storage of case-related documents, cost savings, and better accessibility. The system helps the judiciary stay organized, efficient, and reliable, supporting faster and more transparent legal proceedings.

## **NORMALIZATION:**

### **FIRST NORMAL FORM(1NF):**

Data is stored in rows and columns. Each column has atomic values (e.g., each party has a unique Party\_ID, each lawyer has a Lawyer\_ID).

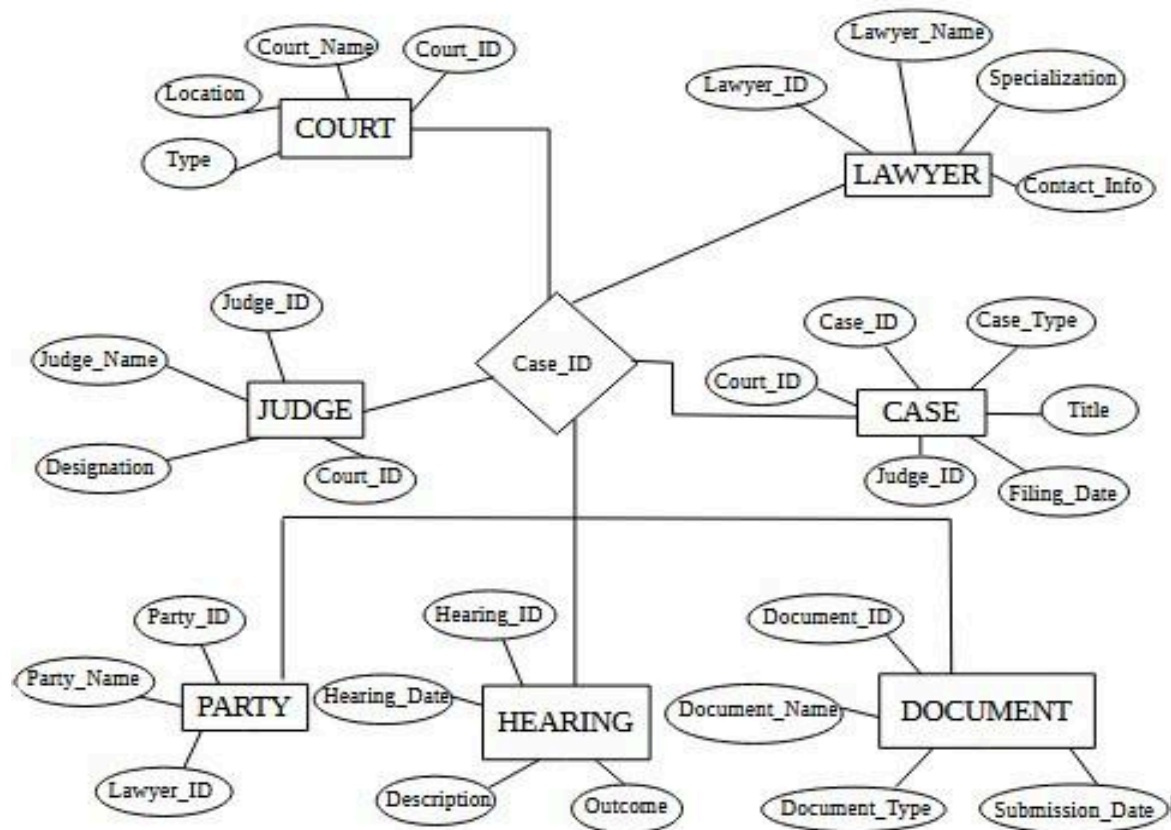
### **SECOND NORMAL FORM(2NF):**

Ensures that all non-key attributes are fully dependent on the primary key. Example: Hearing details are dependent on Hearing\_ID and Case\_ID.

### **THIRD NORMAL FORM(3NF):**

Eliminates transitive dependencies. Example: Lawyer specialization depends only on Lawyer\_ID, not on Case\_ID indirectly.

## ER DIAGRAM:



## TABLE NAME: CASE

COLUMN NAME	DATA TYPE	INDEX	DESCRIPTION
CASE_ID	INT	PRIMARY KEY	Unique identifier for case
TITLE	VARCHAR		Case Title
CASE_TYPE	VARCHAR		Type of the case(Civil, Criminal, Family, etc.)
FILING_DATE	DATE		Date when the case was filed
COURT_ID	INT	FOREIGN KEY	References <b>Court(Court_ID)</b>
JUDGE_ID	INT	FOREIGN KEY	References <b>Judge(Judge_ID)</b>

**TABLE NAME: COURT**

COLUMN NAME	DATA TYPE	INDEX	DESCRIPTION
COURT_ID	INT	PRIMARY KEY	Unique identifier for each court
COURT_NAME	VARCHAR		Name of the court
LOCATION	VARCHAR		Location of the court
TYPE	VARCHAR		Type of the court
CASE_ID	INT	FOREIGN KEY	References <b>Case(Case_ID)</b>

**TABLE NAME: JUDGE**

COLUMN NAME	DATA TYPE	INDEX	DESCRIPTION
JUDGE_ID	INT	PRIMARY KEY	Unique identifier for each judge
JUDGE_NAME	VARCHAR		Name of the court
DESIGNATION	VARCHAR		Designation of judge
COURT_ID	INT	FOREIGN KEY	References <b>Judge(Judge_id)</b>
CASE_ID	INT	FOREIGN KEY	References <b>Case(Case_id)</b>

**TABLE NAME: LAWYER**

COLUMN NAME	DATA TYPE	INDEX	DESCRIPTION
LAWYER_ID	INT	PRIMARY KEY	Unique identifier for each lawyer
LAWYER_NAME	VARCHAR		Name of the lawyer
SPECIALIZATION	VARCHAR		Area of specialization (Criminal, Civil, etc.)
CONTACT_INFO	VARCHAR		Email/phone contact
CASE_ID	INT	FOREIGN KEY	References <b>Case(Case_ID)</b>

**TABLE NAME: PARTY**

COLUMN NAME	DATA TYPE	INDEX	DESCRIPTION
PARTY_ID	INT	PRIMARY KEY	Unique identifier for each party
PARTY_NAME	VARCHAR		Name of the party
ROLE	VARCHAR		Role of the party in the case
LAWYER_ID	INT	FOREIGN KEY	References <b>Lawyer(Lawyer_ID)</b>
CASE_ID	INT	FOREIGN KEY	References <b>Case(Case_ID)</b>

**TABLE NAME: HEARING**

COLUMN NAME	DATA TYPE	INDEX	DESCRIPTION
HEARING_ID	INT	PRIMARY KEY	Unique identifier for each hearing
HEARING_DATE	DATE		Date of hearing
DESCRIPTION	VARCHER		Detailed description of proceedings during the hearing
OUTCOME	VARCHER		Result/decision of the hearing (e.g., Adjourned, Evidence Accepted, Judgment Reserved)
CASE_ID	INT	FOREIGN KEY	References <b>Case(Case_ID)</b>

**TABLE NAME: DOCUMENT**

COLUMN NAME	DATA TYPE	INDEX	DESCRIPTION
DOCUMENT_ID	INT	PRIMARY KEY	Unique identifier for each document
DOCUMENT_NAME	VARCHAR		Name of the document

DOCUMENT_TYPE	VARCHAR		Type of document (FIR, Petition, Evidence, etc.)
SUBMISSION_DATE	DATE		Date document was submitted
CASE_ID	INT	FOREIGN KEY	References Case(Case_ID)

### ★ **Table Creation and Insertion:-**

CREATE TABLE Case ( Case\_ID INT PRIMARY KEY, Title VARCHAR(200), Case\_Type VARCHAR(50), Filing\_Date DATE, Court\_ID INT, Judge\_ID INT);

Table created.

INSERT INTO Case VALUES (301, 'Rajiv vs State of Tamil Nadu', 'Criminal', '10-jan-2020', 201, 401);

1 row created.

INSERT INTO Case VALUES(302, 'Priya vs ABC Corporation', 'Civil', '21-FEB-2021', 202, 402);

1 row created.

INSERT INTO Case VALUES(303, 'State of Maharashtra vs Suresh', 'Criminal', '18-MAR-2021', 203, 403);

1 row created.

INSERT INTO Case VALUES(304, 'Sunil vs Anjali', 'Family', '05-APR-2022', 204, 404);

1 row created.

INSERT INTO Case VALUES(305, 'Bharat Traders vs Esha Ltd', 'Commercial', '20-MAY-2022', 205, 405);

1 row created.

INSERT INTO Case VALUES(306, 'Meena vs Sun Pharma', 'Labor', '11-JUN-2023', 206, 406);

1 row created.

```
INSERT INTO Case VALUES(307, 'Ali vs Fatima', 'Divorce', '01-JUL-2023', 207, 407);
```

1 row created.

```
INSERT INTO Case VALUES(308, 'XYZ Ltd vs Tax Dept.', 'Tax', '15-AUG-2024', 208, 408);
```

1 row created.

```
INSERT INTO Case VALUES(309, 'State of Odisha vs Ramesh', 'Criminal', '07-SEP-2024', 209, 409);
```

1 row created.

```
INSERT INTO Case VALUES(310, 'Kumar vs Kanchalna', 'Family', '22-OCT-2025', 210, 410);
```

1 row created.

```
CREATE TABLE Court (Court_ID INT PRIMARY KEY, Court_Name VARCHAR(100), Location VARCHAR(100), Type VARCHAR(50), Case_ID INT );
```

Table created.

```
INSERT INTO Court VALUES (201, 'Madras High Court', 'Chennai', 'High Court', 301);
```

1 row created.

```
INSERT INTO Court VALUES (202, 'Chennai Civil Court', 'Chennai', 'Civil Court', 302);
```

1 row created.

```
INSERT INTO Court VALUES (203, 'Mumbai Sessions Court', 'Mumbai', 'Sessions Court', 303);
```

1 row created.

```
INSERT INTO Court VALUES (204, 'Delhi Family Court', 'Delhi', 'Family Court', 304);
```

1 row created.

```
INSERT INTO Court VALUES (205, 'Ahmedabad Commercial Court', 'Ahmedabad', 'Commercial Court', 305);
```

1 row created.

```
INSERT INTO Court VALUES (206, 'Pune Labor Court', 'Pune', 'Labor Court', 306);
```

1 row created.

```
INSERT INTO Court VALUES (207, 'Hyderabad Family Court', 'Hyderabad', 'Family Court', 307);
```

1 row created.

```
INSERT INTO Court VALUES (208, 'Kolkata Tax Court', 'Kolkata', 'Tax Court', 308);
```

1 row created.

```
INSERT INTO Court VALUES (209, 'Bhubaneswar Criminal Court', 'Bhubaneswar', 'Criminal Court', 309);
```

1 row created.

```
INSERT INTO Court VALUES (210, 'Coimbatore Family Court', 'Coimbatore', 'Family Court', 310);
```

1 row created.

```
CREATE TABLE Judge (Judge_ID INT PRIMARY KEY, Judge_Name  
VARCHAR(100), Designation VARCHAR(50), Court_ID INT, Case_ID INT);
```

Table created.

```
INSERT INTO Judge VALUES (401, 'Justice R. Subramanian', 'High Court Judge', 201, 301);
```

1 row created.

```
INSERT INTO Judge VALUES (402, 'Justice Priya Menon', 'Civil Judge', 202, 302);
```

1 row created.

```
INSERT INTO Judge VALUES (403, 'Justice Dinesh Patil', 'Sessions Judge', 203, 303);
```

1 row created.



```
INSERT INTO Judge VALUES (404, 'Justice Anita Sharma', 'Family Judge', 204, 304);
```

1 row created.

```
INSERT INTO Judge VALUES (405, 'Justice K. Suresh', 'Commercial Judge', 205, 305);
```

1 row created.

```
INSERT INTO Judge VALUES (406, 'Justice Ramesh Iyer', 'Labor Judge', 206, 306);
```

1 row created.

```
INSERT INTO Judge VALUES (407, 'Justice Leela Devi', 'Family Judge', 207, 307);
```

1 row created.

```
INSERT INTO Judge VALUES (408, 'Justice Aslam Khan', 'Tax Judge', 208, 308);
```

1 row created.

```
INSERT INTO Judge VALUES (409, 'Justice Binod Das', 'Criminal Judge', 209, 309);
```

1 row created.

```
INSERT INTO Judge VALUES (410, 'Justice Kanchana Rao', 'Family Judge', 210, 310);
```

1 row created.

```
CREATE TABLE Lawyer (Lawyer_ID INT PRIMARY KEY, Lawyer_Name  
VARCHAR(100), Specialization VARCHAR(100), Contact_Info VARCHAR(100), Case_ID  
INT);
```

Table created.

```
INSERT INTO Lawyer VALUES (501, 'Adv. Arjun Mehta', 'Criminal Law',  
'arjunmehta@gmail.com', 301);
```

1 row created.

```
INSERT INTO Lawyer VALUES (502, 'Adv. Sneha Nair', 'Civil Litigation',  
'snehanair@lawfirm.com', 302);
```

1 row created.

```
INSERT INTO Lawyer VALUES (503, 'Adv. Ravi Deshmukh', 'Criminal Law',  
'ravideshmukh@courtmail.com', 303);
```

1 row created.

```
INSERT INTO Lawyer VALUES (504, 'Adv. Neha Kapoor', 'Family Law',  
'nehakapoor@lawchambers.com', 304);
```

1 row created.

```
INSERT INTO Lawyer VALUES (505, 'Adv. Rohit Jain', 'Commercial Law',  
'rohitjain@firm.co.in', 305);
```

1 row created.

```
INSERT INTO Lawyer VALUES (506, 'Adv. Deepika Rao', 'Labor Law',  
'deepikarao@legal.org', 306);
```

1 row created.

```
INSERT INTO Lawyer VALUES (507, 'Adv. Imran Siddiqui', 'Family Law',  
'imransiddiqui@law.co', 307);
```

1 row created.

```
INSERT INTO Lawyer VALUES (508, 'Adv. Swati Joshi', 'Tax Law',  
'swatijoshi@taxfirm.com', 308);
```

1 row created.

```
INSERT INTO Lawyer VALUES (509, 'Adv. Raghav Patnaik', 'Criminal Law',  
'raghavp@justice.org', 309);
```

1 row created.

```
INSERT INTO Lawyer VALUES (510, 'Adv. Kiran Bedi', 'Family Law',  
'kiranbedi@familylaw.in', 310);
```

1 row created.

```
CREATE TABLE Party ( Party_ID INT PRIMARY KEY,Party_Name VARCHAR(100),  
Role VARCHAR(50),Lawyer_ID INT,Case_ID INT);
```

Table created.

```
INSERT INTO Party VALUES (601, 'Rajiv', 'Plaintiff', 501, 301);
```

1 row created.

```
INSERT INTO Party VALUES (602, 'State of Tamil Nadu', 'Defendant', 501, 301);
```

1 row created.

```
INSERT INTO Party VALUES (603, 'Priya', 'Plaintiff', 502, 302);
```

1 row created.

```
INSERT INTO Party VALUES (604, 'ABC Corporation', 'Defendant', 502, 302);
```

1 row created.

```
INSERT INTO Party VALUES (605, 'Sunil', 'Petitioner', 504, 304);
```

1 row created.

```
INSERT INTO Party VALUES (606, 'Anjali', 'Respondent', 504, 304);
```

1 row created.

```
INSERT INTO Party VALUES (607, 'Ali', 'Petitioner', 507, 307);
```

1 row created.

```
INSERT INTO Party VALUES (608, 'Fatima', 'Respondent', 507, 307);
```

1 row created.

```
INSERT INTO Party VALUES (609, 'Kumar', 'Petitioner', 510, 310);
```

1 row created.

```
INSERT INTO Party VALUES (610, 'Kanchana', 'Respondent', 510, 310);
```

1 row created.

```
CREATE TABLE Hearing (Hearing_ID INT PRIMARY KEY, Hearing_Date DATE,  
Description VARCHAR2(200), Outcome VARCHAR(200), Case_ID INT);
```

Table created.

```
INSERT INTO Hearing VALUES (701, TO_DATE('15-JAN-2020', 'DD-MON-YYYY'),  
'First hearing - charges framed', 'Adjourned', 301);
```

1 row created.

```
INSERT INTO Hearing VALUES (702, TO_DATE('20-FEB-2021', 'DD-MON-YYYY'),  
'Initial arguments presented', 'Pending', 302);
```

1 row created.

```
INSERT INTO Hearing VALUES (703, TO_DATE('25-MAR-2021', 'DD-MON-YYYY'),  
'Witness statements recorded', 'In Progress', 303);
```

1 row created.

```
INSERT INTO Hearing VALUES (704, TO_DATE('10-APR-2022', 'DD-MON-YYYY'),  
'Counseling session', 'Reconciliation Failed', 304);
```

1 row created.

```
INSERT INTO Hearing VALUES (705, TO_DATE('18-MAY-2022', 'DD-MON-YYYY'),  
'Cross-examination', 'Ongoing', 305);
```

1 row created.

```
INSERT INTO Hearing VALUES (706, TO_DATE('15-JUN-2023', 'DD-MON-YYYY'),  
'Labor dispute negotiation', 'Partial Settlement', 306);
```

1 row created.

```
INSERT INTO Hearing VALUES (707, TO_DATE('05-JUL-2023', 'DD-MON-YYYY'),  
'Final arguments heard', 'Reserved for Judgment', 307);
```

1 row created.

```
INSERT INTO Hearing VALUES (708, TO_DATE('20-AUG-2024', 'DD-MON-YYYY'),  
'Tax documents reviewed', 'Further Hearing Scheduled', 308);
```

1 row created.

```
INSERT INTO Hearing VALUES (709, TO_DATE('12-SEP-2024', 'DD-MON-YYYY'),  
'Evidence submitted by prosecution', 'Adjourned', 309);
```

1 row created.

```
INSERT INTO Hearing VALUES (710, TO_DATE('30-OCT-2025', 'DD-MON-YYYY'),  
'Final order pronounced', 'Judgment Passed', 310);
```

1 row created.

```
CREATE TABLE Document (Document_ID INT PRIMARY KEY, Document_Name  
VARCHAR(100), Document_Type VARCHAR(50), Submission_Date DATE, Case_ID  
INT);
```

Table created.

```
INSERT INTO Document VALUES (801, 'FIR Report', 'Evidence',  
TO_DATE('12-JAN-2020', 'DD-MON-YYYY'), 301);
```

1 row created.

```
INSERT INTO Document VALUES (802, 'Contract Agreement', 'Legal',  
TO_DATE('22-FEB-2021', 'DD-MON-YYYY'), 302);
```

1 row created.

```
INSERT INTO Document VALUES (803, 'Witness Statement', 'Evidence',  
TO_DATE('26-MAR-2021', 'DD-MON-YYYY'), 303);
```

1 row created.

```
INSERT INTO Document VALUES (804, 'Marriage Certificate', 'Personal',  
TO_DATE('12-APR-2022', 'DD-MON-YYYY'), 304);
```

1 row created.

```
INSERT INTO Document VALUES (805, 'Business License', 'Commercial',  
TO_DATE('20-MAY-2022', 'DD-MON-YYYY'), 305);
```

1 row created.

```
INSERT INTO Document VALUES (806, 'Employment Record', 'Labor',  
TO_DATE('17-JUN-2023', 'DD-MON-YYYY'), 306);
```

1 row created.

```
INSERT INTO Document VALUES (807, 'Divorce Petition', 'Legal',  
TO_DATE('07-JUL-2023', 'DD-MON-YYYY'), 307);
```

1 row created.

```
INSERT INTO Document VALUES (808, 'Audit Report', 'Financial',  
TO_DATE('23-AUG-2024', 'DD-MON-YYYY'), 308);
```

1 row created.

```
INSERT INTO Document VALUES (809, 'Forensic Report', 'Evidence',  
TO_DATE('14-SEP-2024', 'DD-MON-YYYY'), 309);
```

1 row created.

```
INSERT INTO Document VALUES (810, 'Custody Agreement', 'Family',  
TO_DATE('01-NOV-2025', 'DD-MON-YYYY'), 310);
```

## ★ CONSTRAINTS COMMANDS

### **Add FOREIGN KEY:**

```
SQL> ALTER TABLE Case ADD CONSTRAINT fk_case_court FOREIGN KEY  
(Court_ID) REFERENCES Court(Court_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Case ADD CONSTRAINT fk_case_judge FOREIGN KEY  
(Judge_ID) REFERENCES Judge(Judge_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Court ADD CONSTRAINT fk_court_case FOREIGN KEY  
(Case_ID) REFERENCES Case(Case_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Judge ADD CONSTRAINT fk_judge_court FOREIGN KEY  
(Court_ID) REFERENCES Court(Court_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Judge ADD CONSTRAINT fk_judge_case FOREIGN KEY  
(Case_ID) REFERENCES Case(Case_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Lawyer ADD CONSTRAINT fk_lawyer_case FOREIGN KEY  
(Case_ID) REFERENCES Case(Case_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Party ADD CONSTRAINT fk_party_lawyer FOREIGN KEY  
(Lawyer_ID) REFERENCES Lawyer(Lawyer_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Party ADD CONSTRAINT fk_party_case FOREIGN KEY  
(Case_ID) REFERENCES Case(Case_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Hearing ADD CONSTRAINT fk_hearing_case FOREIGN KEY  
(Case_ID) REFERENCES Case(Case_ID);
```

TABLE ALTERED

```
SQL> ALTER TABLE Document ADD CONSTRAINT fk_document_case FOREIGN KEY  
(Case_ID) REFERENCES Case(Case_ID);
```

TABLE ALTERED

**NOT NULL:**

```
SQL> ALTER TABLE Judge MODIFY Judge_Name VARCHAR(100) NOT NULL;
```

TABLE ALTERED

**UNIQUE:**

```
SQL> ALTER TABLE Lawyer ADD CONSTRAINT unique_contact UNIQUE  
(Contact_Info);
```

TABLE ALTERED

## **CHECK:**

```
SQL> ALTER TABLE Case ADD CONSTRAINT chk_case_type CHECK (Case_Type IN ('Civil', 'Criminal', 'Family', 'Labor', 'Tax', 'Commercial', 'Divorce'));
```

TABLE ALTERED

## **TCL COMMAND:-**

```
SQL> COMMIT;
```

## **★ DDL COMMANDS - Data Definition Language**

### **Add a New Column to a Table:**

```
SQL> ALTER TABLE Lawyer ADD Bar_Registration_No VARCHAR(20);
```

TABLE ALTERED

### **Rename an Existing Column:**

```
SQL> ALTER TABLE Case RENAME COLUMN Case_Type TO Case_Category;
```

TABLE ALTERED

### **Modify a Column Data Type:**

```
SQL> ALTER TABLE Court MODIFY Location VARCHAR(150);
```

TABLE ALTERED

### **Set a Default Value for a Column:**

```
SQL> ALTER TABLE Judge MODIFY Designation VARCHAR(50) DEFAULT 'Judge';
```

TABLE ALTERED

### **Drop a Column from a Table:**

```
SQL> ALTER TABLE Party DROP COLUMN Role;
```

TABLE DROPPED

### **Drop a Table:**

```
SQL> DROP TABLE Hearing;
```

TABLE DROPPED



**Rename a Table:**

SQL> RENAME Document TO Case\_Documents;  
Table renamed.

**Truncate Table:**

SQL> TRUNCATE Table Court;

Table truncated.

## ★ DML COMMANDS - Data Manipulation Language

**Change judge designation:**

SQL> UPDATE Judge SET Designation = 'Senior Civil Judge' WHERE Judge\_ID = 402;

1 row updated.

**Update hearing outcome:**

SQL> UPDATE Hearing SET Outcome = 'Judgment Reserved' WHERE Hearing\_ID = 702;

1 row updated.

**Correct case title:**

SQL> UPDATE Case SET Title = 'Priya vs ABC Ltd' WHERE Case\_ID = 302;

1 row updated.

**Change court location:**

SQL> UPDATE Court SET Location = 'New Chennai' WHERE Court\_ID = 202;

1 row updated.

**Delete a document:**

SQL> DELETE FROM CASE\_DOCUMENTS WHERE Document\_ID = 810;

1 row deleted.

**View all judges:**

SQL> SELECT \* FROM Judge;

JUDGE_ID	JUDGE_NAME	DESIGNATION	COURT_ID	CASE_ID
401	Justice R. Subramanian	High Court Judge	201	301
402	Justice Priya Menon	Senior Civil Judge	202	302

403	Justice Dinesh Patil	Sessions Judge	203	303
404	Justice Anita Sharma	Family Judge	204	304
405	Justice K. Suresh	Commercial Judge	205	305
406	Justice Ramesh Iyer	Labor Judge	206	306
407	Justice Leela Devi	Family Judge	207	307
408	Justice Aslam Khan	Tax Judge	208	308
409	Justice Binod Das	Criminal Judge	209	309
410	Justice Kanchana Rao	Family Judge	210	310

10 rows selected

**View case details of 'Criminal' type:**

SQL>SELECT \* FROM Case WHERE Case\_Category= 'Criminal';

<b>CASE_ID</b>	<b>TITLE</b>	<b>CASE_CATEGORY</b>	<b>FILING_DATE</b>	<b>COURT_ID</b>	<b>JUDGE_ID</b>
301	Rajiv vs State of Tamil Nadu	Criminal	10-JAN-20	201	401
303	State of Maharashtra vs Suresh	Criminal	18-MAR-21	203	403
309	State of Odisha vs Ramesh	Criminal	07-SEP-24	209	409

3 rows selected

## ★ AGGREGATE FUNCTIONS AND SORTING:-

### 1. COUNT – Total Number of Cases:

SQL> SELECT COUNT(\*) AS Total\_Cases FROM Case;

TOTAL_CASES
10

### 2. MIN / MAX – Earliest and Latest Case Filing Dates:

SQL>SELECT MIN(Filing\_Date) AS Earliest\_Case,MAX(Filing\_Date) AS Latest\_Case  
FROM Case;

EARLIEST_CASE	LATEST_CASE
10-JAN-20	22-OCT-25

### 3. ORDER BY – List Lawyers Alphabetically:

SQL> SELECT Lawyer\_ID, Lawyer\_Name, Specialization FROM Lawyer ORDER BY  
Lawyer\_Name ASC;

LAWYER_ID	LAWYER_NAME	SPECIALIZATION
501	Adv. Arjun Mehta	Criminal Law
506	Adv. Deepika Rao	Labor Law
507	Adv. Imran Siddiqui	Family Law
510	Adv. Kiran Bedi	Family Law
504	Adv. Neha Kapoor	Family Law
509	Adv. Raghav Patnaik	Criminal Law
503	Adv. Ravi Deshmukh	Criminal Law
505	Adv. Rohit Jain	Commercial Law
502	Adv. Sneha Nair	Civil Litigation
508	Adv. Swati Joshi	Tax Law

10 rows selected.

#### 4. SUM/COUNT + GROUP BY – Total Hearings per Case:

SQL> SELECT Case\_ID, COUNT(\*) AS Total\_Hearings FROM Hearing GROUP BY Case\_ID ORDER BY Total\_Hearings DESC;

CASE_ID	TOTAL_HEARINGS
301	1
302	1
303	1
304	1
310	1
306	1
307	1
308	1
309	1
305	1

10 rows selected.

#### 5. AVG of Case\_IDs by Category

SQL> SELECT Case\_Category, AVG(Case\_ID) AS Avg\_CaseID FROM Case GROUP BY Case\_Category;

CASE_CATEGORY	AVG_CASEID
Criminal	304
Civil	302
Family	307
Commercial	305
Labor	306
Divorce	307
Tax	308

7 rows selected

## 6. DESCRIBE THE TABLE STRUCTURE

SQL>DESC JUDGE;

Column Name	Null?	Data Type
JUDGE_ID	NOT NULL	NUMBER(38)
JUDGE_NAME	NOT NULL	VARCHAR2(100)
DESIGNATION		VARCHAR2(50)
COURT_ID		NUMBER(38)
CASE_ID		NUMBER(38)

## 7. COUNT - Total number of cases

SQL>SELECT COUNT(CASE\_ID) "TOTAL CASES" FROM CASE;

TOTAL CASES
10

## 8. Absolute Value

SQL>SELECT ABS(20) "ABSOLUTE VALUE" FROM DUAL;

ABSOLUTE VALUE
20

## 9. ROUND

SQL>SELECT round (2678.96) "round" FROM Dual ;

ROUND
2679

## 10. POWER

SQL>SELECT power (3,2) "power" FROM Dual ;

POWER
9

### 11. SQUARE ROOT

SQL> SELECT sqrt (25) "square root" FROM Dual ;

SQUARE ROOT
5

### 12. EXPONENT

SQL>SELECT exp (5) "exponent" FROM Dual ;

EXPONENT
148.413159

### 13. GREATEST

SQL> SELECT greatest (45643,10458739,235870) "number" FROM Dual ;

NUMBER
10458739

### 14. LEAST

SQL>SELECT least (45643,10458739,235870) "number" FROM Dual ;

ROUND
45643

### 15. MODULOUS

SQL>SELECT mod (15,8) "number" FROM Dual ;

NUMBER
7

### 16. TRUNC

SQL>SELECT trunc (138.356,1) "number" FROM Dual ;

NUMBER
138.3

### 17. FLOOR

SQL>SELECT floor (28.6) "number" FROM Dual ;

NUMBER
28

### 18. CEIL

SQL>SELECT ceil (38.6) "number" FROM Dual ;

NUMBER
39

### 19. LOWERCASE

SQL> SELECT LOWER(JUDGE\_NAME) FROM JUDGE;

LOWER_JUDGE_NAME
justice r. subramanian
justice priya menon
justice dinesh patil
justice anita sharma
justice k. suresh
justice ramesh iyer
justice leela devi
justice aslam khan
justice binod das
justice kanchana rao

10 rows selected.

## 20. INITCAP

SQL> SELECT INITCAP(JUDGE\_NAME) FROM JUDGE;

INITCAP(JUDGE_NAME)
Justice R. Subramanian
Justice Priya Menon
Justice Dinesh Patil
Justice Anita Sharma
Justice K. Suresh
Justice Ramesh Iyer
Justice Leela Devi
Justice Aslam Khan
Justice Binod Das
Justice Kanchana Rao

10 rows selected.

## 21. UPPERCASE

SQL> SELECT UPPER(JUDGE\_NAME) FROM JUDGE;

UPPER(JUDGE_NAME)
JUSTICE R. SUBRAMANIAN
JUSTICE PRIYA MENON
JUSTICE DINESH PATIL
JUSTICE ANITA SHARMA
JUSTICE K. SURESH
JUSTICE RAMESH IYER
JUSTICE LEELA DEVI



JUSTICE ASLAM KHAN
JUSTICE BINOD DAS
JUSTICE KANCHANA RAO

10 rows selected.

## 22. ASCII

SQL> SELECT ASCII('a') FROM DUAL;

<b>ASCII('A')</b>
97

## 23. LENGTH

SQL> SELECT LENGTH('LOCATION') FROM DUAL;

<b>LENGTH('LOCATION')</b>
8

## 24. LTRIM

SQL> SELECT LTRIM('table name','d') FROM DUAL;

<b>LTRIM</b>
table name

## 25. RTRIM

SQL> SELECT RTRIM('table name', 'a') FROM Dual ;

<b>RTRIM</b>
table name

## ★ SET OPERATORS:-

### **UNION:**

SQL>SELECT Case\_ID FROM Case UNION SELECT Case\_ID FROM Lawyer;

CASE_ID
301
302
303
304
305
306
307
308
309
310

10 rows selected.

### **UNION ALL:**

SQL>SELECT Judge\_ID FROM Judge UNION ALL SELECT Judge\_ID FROM Case;

JUDGE_ID
401
402
403
404
405
406

407
408
409
410
401
402
403
404
405
406
407
408
409
410

20 rows selected.

# INTERSECT:

SQL>SELECT Case\_ID FROM Hearing INTERSECT SELECT Case\_ID FROM Document;

CASE_ID
301
302
303
304
305
306

307
308
309

9 rows selected.

## ★ JOINS:-

### 1. INNER JOIN: List all case titles with their court names and judges

SQL>SELECT c.Case\_ID, c.Title, ct.Court\_Name, j.Judge\_Name FROM Case c INNER JOIN Court ct ON c.Court\_ID = ct.Court\_ID INNER JOIN Judge j ON c.Judge\_ID = j.Judge\_ID;

CASE_ID	TITLE	COURT_NAME	JUDGE_NAME
301	Rajiv vs State of Tamil Nadu	Madras High Court	Justice R. Subramanian
302	Priya vs ABC Corporation	Chennai Civil Court	Justice Priya Menon
303	State of Maharashtra vs Suresh	Mumbai Sessions Court	Justice Dinesh Patil
304	Sunil vs Anjali	Delhi Family Court	Justice Anita Sharma
305	Bharat Traders vs Esha Ltd	Ahmedabad Commercial Court	Justice K. Suresh
306	Meena vs Sun Pharma	Pune Labor Court	Justice Ramesh Iyer
307	Ali vs Fatima	Hyderabad Family Court	Justice Leela Devi
308	XYZ Ltd vs Tax Dept.	Kolkata Tax Court	Justice Aslam Khan
309	State of Odisha vs Ramesh	Bhubaneswar Criminal Court	Justice Binod Das
310	Kumar vs Kanchana	Coimbatore Family Court	Justice Kanchana Rao

10 rows selected.

## 2. LEFT JOIN:

SQL>SELECT Case.Case\_ID, Case.Title, Judge.Judge\_Name, Judge. Designation FROM Case LEFT JOIN Judge ON Case.Judge\_ID = Judge.Judge\_ID;

CASE_ID	TITLE	JUDGE_NAME	DESIGNATION
301	Rajiv vs State of Tamil Nadu	Justice R. Subramanian	High Court Judge
302	Priya vs ABC Corporation	Justice Priya Menon	Senior Civil Judge
303	State of Maharashtra vs Suresh	Justice Dinesh Patil	Sessions Judge
304	Sunil vs Anjali	Justice Anita Sharma	Family Judge
305	Bharat Traders vs Esha Ltd	Justice K. Suresh	Commercial Judge
306	Meena vs Sun Pharma	Justice Ramesh Iyer	Labor Judge
307	Ali vs Fatima	Justice Leela Devi	Family Judge
308	XYZ Ltd vs Tax Dept.	Justice Aslam Khan	Tax Judge
309	State of Odisha vs Ramesh	Justice Binod Das	Criminal Judge
310	Kumar vs Kanchana	Justice Kanchana Rao	Family Judge

10 rows selected.

## 3. RIGHT JOIN:

SQL>SELECT l.Lawyer\_Name, l.Specialization, c.Title FROM Lawyer l RIGHT JOIN Case c ON l.Case\_ID = c.Case\_ID;

LAWYER_NAME	SPECIALIZATION	TITLE
Adv. Arjun Mehta	Criminal Law	Rajiv vs State of Tamil Nadu
Adv. Sneha Nair	Civil Litigation	Priya vs ABC Corporation
Adv. Ravi Deshmukh	Criminal Law	State of Maharashtra vs Suresh
Adv. Neha Kapoor	Family Law	Sunil vs Anjali
Adv. Rohit Jain	Commercial Law	Bharat Traders vs Esha Ltd

Adv. Deepika Rao	Labor Law	Meena vs Sun Pharma
Adv. Imran Siddiqui	Family Law	Ali vs Fatima
Adv. Swati Joshi	Tax Law	XYZ Ltd vs Tax Dept.
Adv. Raghav Patnaik	Criminal Law	State of Odisha vs Ramesh
Adv. Kiran Bedi	Family Law	Kumar vs Kanchana

10 rows selected.

#### 4. FULL OUTER JOIN:

SQL> SELECT Case.Case\_ID, Case.Title, Hearing.Hearing\_Date, Hearing.Outcome FROM Case FULL OUTER JOIN Hearing ON Case.Case\_ID = Hearing.Case\_ID;

CASE_ID	TITLE	HEARING_DATE	OUTCOME
301	Rajiv vs State of Tamil Nadu	15-JAN-20	Adjourned
302	Priya vs ABC Corporation	20-FEB-21	Judgment Reserved
303	State of Maharashtra vs Suresh	25-MAR-21	In Progress
304	Sunil vs Anjali	10-APR-22	Reconciliation Failed
305	Bharat Traders vs Esha Ltd	18-MAY-22	Ongoing
306	Meena vs Sun Pharma	15-JUN-23	Partial Settlement
307	Ali vs Fatima	05-JUL-23	Reserved for Judgment
308	XYZ Ltd vs Tax Dept.	20-AUG-24	Further Hearing Scheduled
309	State of Odisha vs Ramesh	12-SEP-24	Adjourned
310	Kumar vs Kanchana	30-OCT-25	Judgment Passed

10 rows selected.

### 5. SELF JOIN: Find pairs of cases handled by the same judge

```
SELECT c1.Case_ID AS Case1, c2.Case_ID AS Case2, j.Judge_Name FROM Case c1 JOIN  
Case c2 ON c1.Judge_ID = c2.Judge_ID AND c1.Case_ID < c2.Case_ID JOIN Judge j ON  
c1.Judge_ID = j.Judge_ID;
```

NO ROWS SELECTED.

### 6. CROSS JOIN: All combinations of lawyers and judges

```
SQL>SELECT CASE.CASE_ID,LAWYER.LAWYER_NAME FROM CASE CROSS JOIN  
LAWYER WHERE CASE.CASE_ID=LAWYER.CASE_ID;
```

CASE_ID	LAWYER_NAME
301	Adv. Arjun Mehta
302	Adv. Sneha Nair
303	Adv. Ravi Deshmukh
304	Adv. Neha Kapoor
305	Adv. Rohit Jain
306	Adv. Deepika Rao
307	Adv. Imran Siddiqui
308	Adv. Swati Joshi
309	Adv. Raghav Patnaik
310	Adv. Kiran Bedi

### 7. JOIN with WHERE :

```
SQL> SELECT Lawyer.Lawyer_Name, Case.Title, Case.Case_Category FROM Lawyer,  
Case WHERE Lawyer.Case_ID = Case.Case_ID;
```

LAWYER_NAME	TITLE	CASE_CATEGORY
Adv. Arjun Mehta	Rajiv vs State of Tamil Nadu	Criminal
Adv. Sneha Nair	Priya vs ABC Corporation	Civil
Adv. Ravi Deshmukh	State of Maharashtra vs Suresh	Criminal

Adv. Neha Kapoor	Sunil vs Anjali	Family
Adv. Rohit Jain	Bharat Traders vs Esha Ltd	Commercial
Adv. Deepika Rao	Meena vs Sun Pharma	Labor
Adv. Imran Siddiqui	Ali vs Fatima	Divorce
Adv. Swati Joshi	XYZ Ltd vs Tax Dept.	Tax
Adv. Raghav Patnaik	State of Odisha vs Ramesh	Criminal
Adv. Kiran Bedi	Kumar vs Kanchana	Family

10 rows selected.

### ★ INTEGRITY CONSTRAINTS:-

SQL>SELECT \* FROM Case;

CASE_ID	TITLE	CASE_TYPE	FILING_DATE	COURT_ID	JUDGE_ID
301	Rajiv vs State of Tamil Nadu	Criminal	10-JAN-20	201	401
302	Priya vs ABC Corporation	Civil	21-FEB-21	202	402
303	State of Maharashtra vs Suresh	Criminal	18-MAR-21	203	403
304	Sunil vs Anjali	Family	05-APR-22	204	404
305	Bharat Traders vs Esha Ltd	Commercial	20-MAY-22	205	405
306	Meena vs Sun Pharma	Labor	11-JUN-23	206	406
307	Ali vs Fatima	Divorce	01-JUL-23	207	407
308	XYZ Ltd vs Tax Dept.	Tax	15-AUG-24	208	408



309	State of Odisha vs Ramesh	Criminal	07-SEP-24	209	409
310	Kumar vs Kanchana	Family	22-OCT-25	210	410

10 rows selected.

SQL> SELECT \* FROM LAWYER;

LAWYER_ID	LAWYER_NAME	SPECIALIZATION	CONTACT_INFO	CASE_ID
501	Adv. Arjun Mehta	Criminal Law	arjunmehta@gmail.com	301
502	Adv. Sneha Nair	Civil Litigation	snehanair@lawfirm.com	302
503	Adv. Ravi Deshmukh	Criminal Law	ravideshmukh@courtmail.com	303
504	Adv. Neha Kapoor	Family Law	nehakapoor@lawchambers.com	304
505	Adv. Rohit Jain	Commercial Law	rohitjain@firm.co.in	305
506	Adv. Deepika Rao	Labor Law	deepikarao@legal.org	306
507	Adv. Imran Siddiqui	Family Law	imransiddiqui@law.com	307
508	Adv. Swati Joshi	Tax Law	swatijoshi@taxfirm.com	308
509	Adv. Raghav Patnaik	Criminal Law	raghavp@justice.org	309
510	Adv. Kiran Bedi	Family Law	kiranbedi@familylaw.in	310

10 rows selected.

SQL> SELECT Lawyer.Lawyer\_ID, Lawyer\_Name, Specialization, Contact\_Info,  
Case.Case\_ID, Title, Case\_Category FROM Lawyer, Case WHERE Lawyer.Case\_ID =  
Case.Case\_ID;

LAWYER_ID	LAWYER_NAME	SPECIALIZATION	CONTACT_INFO	CASE_ID	TITLE	CASE_TYPE
-----------	-------------	----------------	--------------	---------	-------	-----------

501	Adv. Arjun Mehta	Criminal Law	arjunmehta@gmail.com	301	Rajiv vs State of Tamil Nadu	Criminal
502	Adv. Sneha Nair	Civil Litigation	snehanair@lawfirm.com	302	Priya vs ABC Corporation	Civil
503	Adv. Ravi Deshmukh	Criminal Law	ravideshmukh@courtmail.com	303	State of Maharashtra vs Suresh	Criminal
504	Adv. Neha Kapoor	Family Law	nehakapoor@lawchambers.com	304	Sunil vs Anjali	Family
505	Adv. Rohit Jain	Commercial Law	rohitjain@firm.co.in	305	Bharat Traders vs Esha Ltd	Commercial
506	Adv. Deepika Rao	Labor Law	deepikarao@legal.org	306	Meena vs Sun Pharma	Labor
507	Adv. Imran Siddiqui	Family Law	imransiddiqui@law.co	307	Ali vs Fatima	Divorce
508	Adv. Swati Joshi	Tax Law	swatijoshi@taxfirm.com	308	XYZ Ltd vs Tax Dept.	Tax
509	Adv. Raghav Patnaik	Criminal Law	raghavp@justice.org	309	State of Odisha vs Ramesh	Criminal
510	Adv. Kiran Bedi	Family Law	kiranbedi@familylaw.in	310	Kumar vs Kanchana	Family

10 rows selected.

### ★ VIEW:-

#### CREATE VIEW:

```
SQL> CREATE VIEW DOC_DETAIL AS SELECT DOCUMENT_ID,
DOCUMENT_NAME, DOCUMENT_TYPE, CASE_ID FROM DOCUMENTS WHERE
CASE_ID = 304;
```

View created.

```
SQL> CREATE VIEW Case_Judge_View AS SELECT case.Case_ID, Title,  
judge.Judge_Name, Designation FROM Case JOIN Judge ON case.Judge_ID =  
judge.Judge_ID;
```

View created.

```
SQL> SELECT * FROM Case_Judge_View;
```

Case ID	Case Title	Judge Name	Designation
301	Rajiv vs State of Tamil Nadu	Justice R. Subramanian	High Court Judge
302	Priya vs ABC Corporation	Justice Priya Menon	Senior Civil Judge
303	State of Maharashtra vs Suresh	Justice Dinesh Patil	Sessions Judge
304	Sunil vs Anjali	Justice Anita Sharma	Family Judge
305	Bharat Traders vs Esha Ltd	Justice K. Suresh	Commercial Judge
306	Meena vs Sun Pharma	Justice Ramesh Iyer	Labor Judge
307	Ali vs Fatima	Justice Leela Devi	Family Judge
308	XYZ Ltd vs Tax Dept.	Justice Aslam Khan	Tax Judge
309	State of Odisha vs Ramesh	Justice Binod Das	Criminal Judge
310	Kumar vs Kanchana	Justice Kanchana Rao	Family Judge

10 rows selected

**CREATE OR REPLACE VIEW:**

```
SQL> CREATE OR REPLACE VIEW Case_Judge_View AS SELECT CASE.Case_ID,  
Title, JUDGE.JUDGE_ID, Judge_Name, Designation, FROM Case JOIN Judge ON  
CASE.Judge_ID = JUDGE.Judge_ID ;
```

View created.

```
SQL> SELECT * FROM Case_Judge_View;
```

CASE_ID	TITLE	JUDGE_ID	JUDGE_NAME	DESIGNATION
301	Rajiv vs State of Tamil Nadu	401	Justice R. Subramanian	High Court Judge
302	Priya vs ABC Corporation	402	Justice Priya Menon	Senior Civil Judge
303	State of Maharashtra vs Suresh	403	Justice Dinesh Patil	Sessions Judge
304	Sunil vs Anjali	404	Justice Anita Sharma	Family Judge
305	Bharat Traders vs Esha Ltd	405	Justice K. Suresh	Commercial Judge
306	Meena vs Sun Pharma	406	Justice Ramesh Iyer	Labor Judge
307	Ali vs Fatima	407	Justice Leela Devi	Family Judge

308	XYZ Ltd vs Tax Dept.	408	Justice Aslam Khan	Tax Judge
309	State of Odisha vs Ramesh	409	Justice Binod Das	Criminal Judge
310	Kumar vs Kanchana	410	Justice Kanchana Rao	Family Judge

10 rows selected.

#### **UPDATE VALUES IN VIEW:**

```
SQL> UPDATE Case_Judge_View SET JUDGE_NAME='Justice R. Subramanian' WHERE CASE_ID=303;
```

1 row updated.

#### **DROP VIEW:**

```
SQL> DROP VIEW Case_Judge_View;
```

View dropped.

### **★ TABLE AND RECORD:-**

#### **1. HEARING TABLE**

```
SQL> DECLARE
2  hearing_rec Hearing%ROWTYPE;
3 BEGIN
4  SELECT * INTO hearing_rec
5  FROM Hearing
6  WHERE Hearing_ID = 701;
7  DBMS_OUTPUT.PUT_LINE('Hearing ID: ' || hearing_rec.Hearing_ID);
8  DBMS_OUTPUT.PUT_LINE('Date: ' || TO_CHAR(hearing_rec.Hearing_Date,
'DD-MON-YYYY'));
10 DBMS_OUTPUT.PUT_LINE('Description: ' || hearing_rec.Description);
11 DBMS_OUTPUT.PUT_LINE('Outcome      : ' || hearing_rec.Outcome);
12 DBMS_OUTPUT.PUT_LINE('Case ID       : ' || hearing_rec.Case_ID);
```

13 END;

14 /

### OUTPUT:

Hearing ID : 701

Date : 15-JAN-2020

Description : First hearing - charges framed

Outcome : Adjourned

Case ID : 301

PL/SQL procedure successfully completed.

## 2. CASE TABLE

SQL> DECLARE

2 TYPE lawyer\_rec\_type IS RECORD (

3 lawyer\_id INT,

4 name VARCHAR2(100),

5 specialization VARCHAR2(100),

6 case\_id INT

7 );

8

9 lawyer1 lawyer\_rec\_type;

10 lawyer2 lawyer\_rec\_type;

11 BEGIN

12

13 lawyer1.lawyer\_id := 501;

14 lawyer1.name := 'Adv. Arjun Mehta';

15 lawyer1.specialization := 'Criminal Law';

16 lawyer1.case\_id := 301;

17

18 lawyer2.lawyer\_id := 502;

19 lawyer2.name := 'Adv. Sneha Nair';

20 lawyer2.specialization := 'Civil Litigation';

21 lawyer2.case\_id := 302;

22

23 -- Display

24 DBMS\_OUTPUT.PUT\_LINE('Lawyer1: ' || lawyer1.name || ', Spec: ' ||  
lawyer1.specialization || ', Case ID: ' || lawyer1.case\_id);

25 DBMS\_OUTPUT.PUT\_LINE('Lawyer2: ' || lawyer2.name || ', Spec: ' ||  
lawyer2.specialization || ', Case ID: ' || lawyer2.case\_id);

26 END;

27 /

**OUTPUT:**

Lawyer1: Adv. Arjun Mehta, Spec: Criminal Law, Case ID: 301

Lawyer2: Adv. Sneha Nair, Spec: Civil Litigation, Case ID: 302

PL/SQL procedure successfully completed.

**★ FUNCTIONS:-****1. GET JUDGE NAME BY CASE ID:**

```
SQL> CREATE OR REPLACE FUNCTION Get_Judge_Name(p_case_id IN NUMBER)
 2 RETURN VARCHAR2
 3 IS
 4   v_judge_name VARCHAR2(100);
 5 BEGIN
 6   SELECT Judge_Name
 7   INTO v_judge_name
 8   FROM Judge
 9   WHERE Case_ID = p_case_id;
10
11   RETURN v_judge_name;
12 EXCEPTION
13   WHEN NO_DATA_FOUND THEN
14     RETURN 'No Judge Assigned';
15 END;
16 /
```

Function created.

```
SQL> DECLARE
 2   input_case_id NUMBER;
 3   result VARCHAR2(100);
 4 BEGIN
 5   input_case_id := &enter_case_id;
 6   result := Get_Judge_Name(input_case_id);
 7
 8   DBMS_OUTPUT.PUT_LINE('The Judge for Case ID ' || input_case_id || ' is: ' || result);
 9 END;
10 /
```

**OUTPUT:**

Enter value for enter\_case\_id: 305

old 5: input\_case\_id := &enter\_case\_id;

```
new 5: input_case_id := 305;
The Judge for Case ID 305 is: Justice K. Suresh
```

PL/SQL procedure successfully completed.

## 2. GET LAWYER SPECIALIZATION USING LAWYER\_ID:

```
SQL> CREATE OR REPLACE FUNCTION Get_Lawyer_Specialization(p_lawyer_id IN
NUMBER)
2 RETURN VARCHAR2
3 IS
4   v_spec VARCHAR2(100);
5 BEGIN
6   SELECT Specialization
7   INTO v_spec
8   FROM Lawyer
9   WHERE Lawyer_ID = p_lawyer_id;
10
11  RETURN v_spec;
12 EXCEPTION
13  WHEN NO_DATA_FOUND THEN
14    RETURN 'No Lawyer Found';
15 END;
16 /
```

Function created.

```
SQL> DECLARE
2   input_lawyer_id NUMBER;
3   result VARCHAR2(100);
4 BEGIN
5   input_lawyer_id := &enter_lawyer_id;
6   result := Get_Lawyer_Specialization(input_lawyer_id);
7
8   DBMS_OUTPUT.PUT_LINE('The Specialization of Lawyer ID ' || input_lawyer_id || '
is: ' || result);
9 END;
10 /
```

### OUTPUT:

```
Enter value for enter_lawyer_id: 505
old 5: input_lawyer_id := &enter_lawyer_id;
new 5: input_lawyer_id := 505;
The Specialization of Lawyer ID 505 is: Commercial Law
```



PL/SQL procedure successfully completed.

### 3. GET CASE TITLE USING JUDGE\_ID:

```
SQL> CREATE OR REPLACE FUNCTION Get_Case_Title_By_Judge(p_judge_id IN  
NUMBER)
```

```
2 RETURN VARCHAR2  
3 IS  
4   v_title VARCHAR2(200);  
5 BEGIN  
6   SELECT c.Title  
7   INTO v_title  
8   FROM Case c  
9   JOIN Judge j ON c.Case_ID = j.Case_ID  
10  WHERE j.Judge_ID = p_judge_id;  
11  
12  RETURN v_title;  
13 EXCEPTION  
14  WHEN NO_DATA_FOUND THEN  
15    RETURN 'No Case Found for this Judge';  
16 END;  
17 /
```

Function created.

```
SQL> DECLARE
```

```
2   input_judge_id NUMBER;  
3   result VARCHAR2(200);  
4 BEGIN  
5   input_judge_id := &enter_judge_id;  
6   result := Get_Case_Title_By_Judge(input_judge_id);  
7  
8   DBMS_OUTPUT.PUT_LINE('The Case handled by Judge ID ' || input_judge_id || ' is: '  
|| result);  
9 END;  
10 /
```

### OUTPUT:

Enter value for enter\_judge\_id: 401

old 5: input\_judge\_id := &enter\_judge\_id;

new 5: input\_judge\_id := 401;

The Case handled by Judge ID 401 is: Rajiv vs State of Tamil Nadu

PL/SQL procedure successfully completed.

## ★ PACKAGES:-

### 1. PACKAGE TO GET HEARING OUTCOME AND COURT INFO:

```
SQL> CREATE OR REPLACE PACKAGE Case_Hearing_Pkg AS
  2  FUNCTION Get_Latest_Hearing_Outcome(p_case_id IN NUMBER) RETURN
VARCHAR2;
  3  FUNCTION Get_Court_Name(p_case_id IN NUMBER) RETURN VARCHAR2;
  4  END Case_Hearing_Pkg;
  5  /
```

Package created.

```
SQL> CREATE OR REPLACE PACKAGE BODY Case_Hearing_Pkg AS
  2
  3  FUNCTION Get_Latest_Hearing_Outcome(p_case_id IN NUMBER) RETURN
VARCHAR2 IS
  4  v_outcome VARCHAR2(200);
  5  BEGIN
  6  SELECT Outcome INTO v_outcome
  7  FROM (SELECT Outcome FROM Hearing WHERE Case_ID = p_case_id ORDER
BY Hearing_Date DESC)
  8  WHERE ROWNUM = 1;
  9  RETURN v_outcome;
  10 EXCEPTION
  11 WHEN NO_DATA_FOUND THEN
  12 RETURN 'No Hearing Found';
  13 END Get_Latest_Hearing_Outcome;
  14
  15 FUNCTION Get_Court_Name(p_case_id IN NUMBER) RETURN VARCHAR2 IS
  16 v_court VARCHAR2(100);
  17 BEGIN
  18 SELECT Court_Name INTO v_court
  19 FROM Court
  20 WHERE Case_ID = p_case_id;
  21 RETURN v_court;
  22 EXCEPTION
  23 WHEN NO_DATA_FOUND THEN
  24 RETURN 'No Court Found';
  25 END Get_Court_Name;
  26 END Case_Hearing_Pkg;
  27  /
```

Package body created.

```
SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
  2  v_outcome VARCHAR2(200);
  3  v_court   VARCHAR2(100);
  4  input_id  NUMBER := &enter_case_id;
  5 BEGIN
  6  v_outcome := Case_Hearing_Pkg.Get_Latest_Hearing_Outcome(input_id);
  7  v_court   := Case_Hearing_Pkg.Get_Court_Name(input_id);
  8
  9  DBMS_OUTPUT.PUT_LINE('Court Name: ' || v_court);
 10  DBMS_OUTPUT.PUT_LINE('Latest Hearing Outcome: ' || v_outcome);
 11 END;
 12 /
```

#### **OUTPUT:**

```
Enter value for enter_case_id: 302
old 4: input_id NUMBER := &enter_case_id;
new 4: input_id NUMBER := 302;
Court Name: Chennai Civil Court
Latest Hearing Outcome: Judgment Reserved
```

PL/SQL procedure successfully completed.

## **2. CASE AND LAWYER INFO:**

```
SQL> CREATE OR REPLACE PACKAGE Small_Case_Pkg AS
  2  FUNCTION Get_Case_Lawyer_Info(p_case_id IN NUMBER) RETURN VARCHAR2;
  3 END Small_Case_Pkg;
  4 /
```

Package created.

```
SQL> CREATE OR REPLACE PACKAGE BODY Small_Case_Pkg AS
  2
  3  FUNCTION Get_Case_Lawyer_Info(p_case_id IN NUMBER) RETURN VARCHAR2
  IS
  4  v_title VARCHAR2(200);
  5  v_lawyer VARCHAR2(100);
  6 BEGIN
  7  SELECT Title, Lawyer_Name
  8  INTO v_title, v_lawyer
  9  FROM Case c
```

```

10 JOIN Lawyer l ON c.Case_ID = l.Case_ID
11 WHERE c.Case_ID = p_case_id;
12
13 RETURN 'Case: ' || v_title || ', Lawyer: ' || v_lawyer;
14 EXCEPTION
15 WHEN NO_DATA_FOUND THEN
16 RETURN 'No Case or Lawyer Found';
17 END Get_Case_Lawyer_Info;
18
19 END Small_Case_Pkg;
20 /

```

Package body created.

```

SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
2  v_info VARCHAR2(300);
3  input_id NUMBER := &enter_case_id;
4 BEGIN
5  v_info := Small_Case_Pkg.Get_Case_Lawyer_Info(input_id);
6  DBMS_OUTPUT.PUT_LINE(v_info);
7 END;
8 /

```

### OUTPUT:

```

Enter value for enter_case_id: 309
old 3: input_id NUMBER := &enter_case_id;
new 3: input_id NUMBER := 309;
Case: State of Odisha vs Ramesh, Lawyer: Adv. Raghav Patnaik

```

PL/SQL procedure successfully completed.

## ★ CURSORS

### 1. LIST ALL JUDGES:

```

SQL> DECLARE
2  CURSOR judge_cur IS
3  SELECT judge_id, judge_name, designation
4  FROM judge;
5  DECLARE
6  v_id judge.judge_id%TYPE;
7  v_name judge.judge_name%TYPE;

```

```

8  v_desg judge.designation%TYPE;
9  BEGIN
10  OPEN judge_cur;
11  LOOP
12      FETCH judge_cur INTO v_id, v_name, v_desg;
13      EXIT WHEN judge_cur%NOTFOUND;
14
15      DBMS_OUTPUT.PUT_LINE('Judge ID: ' || v_id ||
16                          ', Name: ' || v_name ||
17                          ', Designation: ' || v_desg);
18  END LOOP;
19  CLOSE judge_cur;
20 END;
21 /

```

### OUTPUT:

Judge ID: 401, Name: Justice R. Subramanian, Designation: High Court Judge  
 Judge ID: 402, Name: Justice Priya Menon, Designation: Senior Civil Judge  
 Judge ID: 403, Name: Justice Dinesh Patil, Designation: Sessions Judge  
 Judge ID: 404, Name: Justice Anita Sharma, Designation: Family Judge  
 Judge ID: 405, Name: Justice K. Suresh, Designation: Commercial Judge  
 Judge ID: 406, Name: Justice Ramesh Iyer, Designation: Labor Judge  
 Judge ID: 407, Name: Justice Leela Devi, Designation: Family Judge  
 Judge ID: 408, Name: Justice Aslam Khan, Designation: Tax Judge  
 Judge ID: 409, Name: Justice Binod Das, Designation: Criminal Judge  
 Judge ID: 410, Name: Justice Kanchana Rao, Designation: Family Judge

PL/SQL procedure successfully completed.

## 2. CURSOR WITH JOIN - LAWYERS AND PARTIES:

```

SQL> DECLARE
2  CURSOR party_lawyer_cur IS
3      SELECT p.party_name, p.party_id, l.lawyer_name, l.specialization
4      FROM party p
5      JOIN lawyer l ON p.lawyer_id = l.lawyer_id;
6  BEGIN
7      FOR rec IN party_lawyer_cur LOOP
8          DBMS_OUTPUT.PUT_LINE('Party: ' || rec.party_name ||
9                              ' | Lawyer: ' || rec.lawyer_name ||
10                             ' ( ' || rec.specialization || ')');
11  END LOOP;
12 END;
13 /

```

**OUTPUT:**

Party: Rajiv | Lawyer: Adv. Arjun Mehta (Criminal Law)  
Party: State of Tamil Nadu | Lawyer: Adv. Arjun Mehta (Criminal Law)  
Party: Priya | Lawyer: Adv. Sneha Nair (Civil Litigation)  
Party: ABC Corporation | Lawyer: Adv. Sneha Nair (Civil Litigation)  
Party: Sunil | Lawyer: Adv. Neha Kapoor (Family Law)  
Party: Anjali | Lawyer: Adv. Neha Kapoor (Family Law)  
Party: Ali | Lawyer: Adv. Imran Siddiqui (Family Law)  
Party: Fatima | Lawyer: Adv. Imran Siddiqui (Family Law)  
Party: Kumar | Lawyer: Adv. Kiran Bedi (Family Law)  
Party: Kanchana | Lawyer: Adv. Kiran Bedi (Family Law)

PL/SQL procedure successfully completed.

**3. CURSOR WITH COURT DETAILS:**

```
SQL> DECLARE
2  CURSOR court_cur IS
3      SELECT court_id, court_name, location, type
4      FROM court;
5
6 BEGIN
7  FOR rec IN court_cur LOOP
8      DBMS_OUTPUT.PUT_LINE('Court: ' || rec.court_name ||
9                          ' | Location: ' || rec.location ||
10                         ' | Type: ' || rec.type);
11  END LOOP;
12 END;
13 /
```

**OUTPUT:**

Court: Madras High Court | Location: Chennai | Type: High Court  
Court: Chennai Civil Court | Location: Chennai | Type: Civil Court  
Court: Mumbai Sessions Court | Location: Mumbai | Type: Sessions Court  
Court: Delhi Family Court | Location: Delhi | Type: Family Court  
Court: Ahmedabad Commercial Court | Location: Ahmedabad | Type: Commercial Court  
Court: Pune Labor Court | Location: Pune | Type: Labor Court  
Court: Hyderabad Family Court | Location: Hyderabad | Type: Family Court  
Court: Kolkata Tax Court | Location: Kolkata | Type: Tax Court  
Court: Bhubaneswar Criminal Court | Location: Bhubaneswar | Type: Criminal Court  
Court: Coimbatore Family Court | Location: Coimbatore | Type: Family Court

PL/SQL procedure successfully completed.

#### 4. COUNT HEARINGS FETCHED:

SQL> DECLARE

```
2  CURSOR hearing_cur IS
3      SELECT hearing_id, hearing_date, outcome FROM hearing;
4
5  v_id   hearing.hearing_id%TYPE;
6  v_date hearing.hearing_date%TYPE;
7  v_out  hearing.outcome%TYPE;
8  BEGIN
9      OPEN hearing_cur;
10     LOOP
11         FETCH hearing_cur INTO v_id, v_date, v_out;
12         EXIT WHEN hearing_cur%NOTFOUND;
13
14         DBMS_OUTPUT.PUT_LINE('Fetched Hearing ' || hearing_cur%ROWCOUNT ||
15                               ': ID=' || v_id || ', Outcome=' || v_out);
16     END LOOP;
17     CLOSE hearing_cur;
18 END;
19 /
```

#### OUTPUT:

Fetched Hearing 1: ID=701, Outcome=Adjourned  
Fetched Hearing 2: ID=702, Outcome=Judgment Reserved  
Fetched Hearing 3: ID=703, Outcome=In Progress  
Fetched Hearing 4: ID=704, Outcome=Reconciliation Failed  
Fetched Hearing 5: ID=705, Outcome=Ongoing  
Fetched Hearing 6: ID=706, Outcome=Partial Settlement  
Fetched Hearing 7: ID=707, Outcome=Reserved for Judgment  
Fetched Hearing 8: ID=708, Outcome=Further Hearing Scheduled  
Fetched Hearing 9: ID=709, Outcome=Adjourned  
Fetched Hearing 10: ID=710, Outcome=Judgment Passed  
Fetched Hearing 11: ID=711, Outcome=Pending

PL/SQL procedure successfully completed.

## ★ TRIGGERS

### 1. TRIGGER - LOG HEARING INSERTIONS:

```
SQL> CREATE OR REPLACE TRIGGER trg_hearing_log
  2 AFTER INSERT ON Hearing
  3 FOR EACH ROW
  4 BEGIN
  5   DBMS_OUTPUT.PUT_LINE('New Hearing Added: ' || :NEW.Hearing_ID ||
  6                       ' on ' || TO_CHAR(:NEW.Hearing_Date,'DD-MON-YYYY') ||
  7                       ' for Case ' || :NEW.Case_ID);
  8 END;
  9 /
```

Trigger created.

```
SQL> INSERT INTO Hearing VALUES (711, TO_DATE
('12-DEC-2025','DD-MON-YYYY'), 'Additional arguments', 'Pending', 301);
```

#### OUTPUT:

New Hearing Added: 711 on 12-DEC-2025 for Case 301  
1 row created.

### 2. PREVENT DELETING A CASE IF HEARINGS EXIST:

```
SQL> CREATE OR REPLACE TRIGGER trg_no_case_delete
  2 BEFORE DELETE ON case
  3 FOR EACH ROW
  4 DECLARE
  5   v_count NUMBER;
  6 BEGIN
  7   SELECT COUNT(*) INTO v_count FROM hearing WHERE case_id = :OLD.case_id;
  8   IF v_count > 0 THEN
  9     RAISE_APPLICATION_ERROR(-20001, 'Cannot delete case with hearings.');
```

Trigger created.

```
SQL> DELETE FROM case WHERE case_id = 301;
```

**OUTPUT:** Cannot delete case with hearings.



## ★ EXCEPTION HANDLING

### 1. HANDLE NO DATA FOUND (CASE NOT AVAILABLE):

```
SQL> DECLARE
 2  v_title VARCHAR2(200);
 3 BEGIN
 4  SELECT title
 5  INTO v_title
 6  FROM case -- no quotes needed
 7  WHERE case_id = 999;  8
 9  DBMS_OUTPUT.PUT_LINE('Case Title: ' || v_title);
10
11 EXCEPTION
12  WHEN NO_DATA_FOUND THEN
13    DBMS_OUTPUT.PUT_LINE('No Case found with given ID. ');
14  WHEN OTHERS THEN
15    DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
16 END;
17 /
```

#### OUTPUT:

No Case found with given ID.

PL/SQL procedure successfully completed.

### 2. HANDLE DUPLICATE INSERT INTO JUDGE TABLE:

```
SQL> DECLARE
 2 BEGIN
 3  INSERT INTO judge (judge_id, judge_name, designation, court_id, case_id)
 4  VALUES (401, 'Justice Kumar', 'High Court Judge', 201, 301);
 5  DBMS_OUTPUT.PUT_LINE('Judge Inserted Successfully');
 6 EXCEPTION
 7  WHEN DUP_VAL_ON_INDEX THEN
 8    DBMS_OUTPUT.PUT_LINE(' Duplicate Judge ID not allowed. ');
 9  WHEN OTHERS THEN
10    DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
11 END;
12 /
```

#### OUTPUT:

Duplicate Judge ID not allowed.

PL/SQL procedure successfully completed.

### 3. PREVENT LAWYER WITHOUT CASE:

```
SQL> DECLARE
2   e_no_case EXCEPTION;
3   v_case_id NUMBER := NULL; -- Example missing case
4 BEGIN
5   IF v_case_id IS NULL THEN
6     RAISE e_no_case;
7   END IF;
8
9   INSERT INTO lawyer (lawyer_id, lawyer_name, specialization, contact_info, case_id,
bar_registration_no)
10  VALUES (601, 'Adv. Sharma', 'Criminal Law', '9876543210', v_case_id, 'TN12345');
11
12  DBMS_OUTPUT.PUT_LINE(' Lawyer Inserted Successfully');
13
14 EXCEPTION
15  WHEN e_no_case THEN
16    DBMS_OUTPUT.PUT_LINE('Lawyer must be assigned to a case. ');
17  WHEN OTHERS THEN
18    DBMS_OUTPUT.PUT_LINE(' Error: ' || SQLERRM);
19 END;
20 /
```

#### OUTPUT:

Lawyer must be assigned to a case.

PL/SQL procedure successfully completed.