📓 MySQL CASTING Notes (Deep Practical Guide)

✅ Step 1: Create Database

```
CREATE DATABASE student_casting;
USE student_casting;
```

✅ Step 2: Create Table (With VARCHAR Types)

```
CREATE TABLE students (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    marks VARCHAR(10),          -- String format (e.g. '85', '75.5')
    admission_date VARCHAR(20) -- String format (e.g. '2024-06-01')
);
```

✅ Step 3: Insert Sample Data

```
INSERT INTO students (name, marks, admission_date) VALUES
('Aakash', '85', '2024-06-01'),
('Bhavna', '92', '2024-06-02'),
('Chetan', '75.5', '2024-06-03'),
('Divya', '66.75', '2024-06-04'),
('Ekta', '90', '2024-06-05');
```

🎯 Step 4: Casting Queries with OUTPUT

🔹 1. String to Number (UNSIGNED)

```
SELECT name, marks, CAST(marks AS UNSIGNED) AS marks_number FROM students;
```

📊 Output:

name	marks	marks_number
Aakash	85	85
Bhavna	92	92

name	marks	marks_number
Chetan	75.5	75
Divya	66.75	66
Ekta	90	90

📝 **Note:** Decimal values ka **only integer part** liya jaata hai in `UNSIGNED`.

◈ 2. String to Decimal (5,2)

```
SELECT name, marks, CAST(marks AS DECIMAL(5,2)) AS decimal_marks FROM
students;
```

📊 Output:

name	marks	decimal_marks
Aakash	85	85.00
Bhavna	92	92.00
Chetan	75.5	75.50
Divya	66.75	66.75
Ekta	90	90.00

📝 **Note:** Decimal casting maintains full decimal value (2 digits after point).

◈ 3. String to Date

```
SELECT name, admission_date, CAST(admission_date AS DATE) AS proper_date
FROM students;
```

📊 Output:

name	admission_date	proper_date
Aakash	2024-06-01	2024-06-01
Bhavna	2024-06-02	2024-06-02
Chetan	2024-06-03	2024-06-03
Divya	2024-06-04	2024-06-04
Ekta	2024-06-05	2024-06-05

📝 **Note:** Date casting useful for sorting & filtering by date.

◆ 4. Add 10 to Marks (Arithmetic after Casting)

```
SELECT name, CAST(marks AS UNSIGNED) + 10 AS updated_marks FROM students;
```

📊 Output:

name	updated_marks
Aakash	95
Bhavna	102
Chetan	85
Divya	76
Ekta	100

📝 **Note:** String to number cast is necessary for arithmetic operations.

◆ 5. Show Topper Students (90+ Marks)

```
SELECT name, CAST(marks AS UNSIGNED) AS real_marks
FROM students
WHERE CAST(marks AS UNSIGNED) >= 90;
```

📊 Output:

name	real_marks
Bhavna	92
Ekta	90

📝 **Note:** Filtering will **fail without casting**, as string '85' < '9' in ASCII logic.

📌 Bonus: CONVERT() Function (Same as CAST)

```
SELECT name, CONVERT(marks, DECIMAL(5,2)) AS converted_marks FROM students;
```

📊 Output same as CAST with `DECIMAL`.

✍ Use of CASTING in MySQL

Use Case	Description
Convert string to number/date	For calculations and comparisons

Use Case	Description
Sorting with correct data types	'100' < '9' problem solved with cast
Date formatting	Use CAST for valid DATE column filtering
Arithmetic operations	Add, subtract after number casting
Better data control	Avoid wrong results in WHERE clauses

---

2 topics —

✅ CONCATENATION
✅ NUMERIC FUNCTIONS
ko **Deep Book Format** me likh raha hoon:
Input Table → Query → Output → Matlab/Samjhaav → Note

📓 MySQL Deep Notes (Part 1)

Topics:

1. ✅ CONCATENATION
2. ✅ NUMERIC FUNCTIONS

✅ Step 1: Database and Table Setup

```
CREATE DATABASE student_function;
USE student_function;

CREATE TABLE students (
    id INT AUTO_INCREMENT PRIMARY KEY,
    fname VARCHAR(50),
    lname VARCHAR(50),
    marks VARCHAR(10)
);
```

✅ Step 2: Insert Sample Data

```
INSERT INTO students (fname, lname, marks) VALUES
('Aakash', 'Singh', '85.5'),
('Bhavna', 'Sharma', '92'),
('Chetan', 'Kumar', NULL),
```

```
('Divya', 'Verma', '66.75'),
('Ekta', NULL, '90');
```

◆ 1. CONCATENATION (नाम जोड़ना)

🎯 Objective:

`fname` aur `lname` ko जोड़कर पूरा नाम banana.

🔍 Before Table:

fname	lname
Aakash	Singh
Bhavna	Sharma
Chetan	Kumar
Divya	Verma
Ekta	NULL

✅ Query 1: Basic CONCAT (Risky)

```
SELECT CONCAT(fname, ' ', lname) AS full_name FROM students;
```

📤 Output:

full_name
Aakash Singh
Bhavna Sharma
Chetan Kumar
Divya Verma
NULL

❗ Explanation:

- `CONCAT()` function joins values.
- If **any** part is `NULL`, whole result becomes `NULL`.

✅ Query 2: Safe Version using `CONCAT_WS()`

```
SELECT CONCAT_WS(' ', fname, lname) AS full_name_safe FROM students;
```

📤 Output:

full_name_safe
Aakash Singh
Bhavna Sharma
Chetan Kumar
Divya Verma
Ekta

📝 Note:

- `CONCAT_WS()` = **With Separator** (space in this case).
- It **ignores NULL** values.
- Best for **real-world names**, addresses, or combining strings.

◆ 2. NUMERIC FUNCTIONS (अंक आधारित कार्य)

🎯 Objective:

`marks` column me string format me numbers hain. Us par mathematical operations karna hai.

🔍 Before Table:

fname	marks
Aakash	85.5
Bhavna	92
Chetan	NULL
Divya	66.75
Ekta	90

✅ Query 1: ROUND (निकटतम पूर्णांक)

```
SELECT fname, marks,
  ROUND(CAST(marks AS DECIMAL(5,2))) AS rounded_marks
FROM students;
```

📤 Output:

fname	marks	rounded_marks
Aakash	85.5	86
Bhavna	92	92
Chetan	NULL	NULL
Divya	66.75	67
Ekta	90	90

🔍 Note:

- `ROUND()` rounds decimal to nearest whole number.
- `CAST()` used to convert string → number.

✅ Query 2: FLOOR & CEIL

```
SELECT fname,
  FLOOR(CAST(marks AS DECIMAL(5,2))) AS floored,
  CEIL(CAST(marks AS DECIMAL(5,2))) AS ceiled
FROM students;
```

📤 Output:

fname	floored	ceiled
Aakash	85	86
Bhavna	92	92
Divya	66	67
Ekta	90	90

🔍 Explanation:

- `FLOOR()` = neeche round karega (85.9 → 85)
- `CEIL()` = upar round karega (85.1 → 86)

✅ Query 3: MOD() & ABS()

```
SELECT fname,
  MOD(CAST(marks AS UNSIGNED), 10) AS mod_result,
```

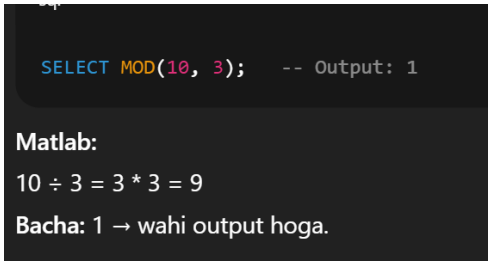
```
  ABS(-CAST(marks AS DECIMAL(5,2))) AS abs_val
FROM students;
```

📤 Output:

fname	mod_result	abs_val
Aakash	5	85.50
Bhavna	2	92.00
Divya	6	66.75
Ekta	0	90.00

🧠 Meaning:

- `MOD()` → Divide karke **remainder** deta hai (85.5 → 85 → mod 10 = 5)
- `ABS()` → Negative ko positive banata hai.



📘 MySQL Deep Notes (Part 2)

Topics:

✅ COALESCE
✅ DATE/TIME
✅ INTERVALS

Jaise tune kaha tha — same **Deep Style: Input Table → Query → Output → Explanation** format me.

📘 MySQL Deep Notes — Part 2 (With Table + Insert + Output)

Topics:

✅ Step 1: Create Table (Full Table for Part 2)

```
CREATE DATABASE student_part2;
USE student_part2;

CREATE TABLE students (
    id INT AUTO_INCREMENT PRIMARY KEY,
    fname VARCHAR(50),
    lname VARCHAR(50),
    marks VARCHAR(10),
    admission_date VARCHAR(20),  -- stored as string for casting
    dob DATE                     -- real DATE type
);
```

✅ Step 2: Insert Data

```
INSERT INTO students (fname, lname, marks, admission_date, dob) VALUES
('Aakash', 'Singh', '85.5', '2024-06-01', '2003-05-20'),
('Bhavna', 'Sharma', '92', '2024-06-02', '2004-03-15'),
('Chetan', 'Kumar', NULL, '2024-06-03', '2002-12-10'),
('Divya', 'Verma', '66.75', '2024-06-04', '2003-08-09'),
('Ekta', NULL, '90', '2024-06-05', '2005-01-01');
```

◆ 1. COALESCE() — NULL ko replace karna

✅ Query

```
SELECT fname,
  COALESCE(marks, '0') AS safe_marks,
  COALESCE(lname, 'Not Given') AS safe_lname
FROM students;
```

📤 Output:

fname	safe_marks	safe_lname
Aakash	85.5	Singh
Bhavna	92	Sharma
Chetan	0	Kumar
Divya	66.75	Verma
Ekta	90	Not Given

□ Explanation:

- `COALESCE()` returns **first NOT NULL** value.
- Real use: Jab kisi value ka fallback/default dena ho (like "Not Given", "0", etc).

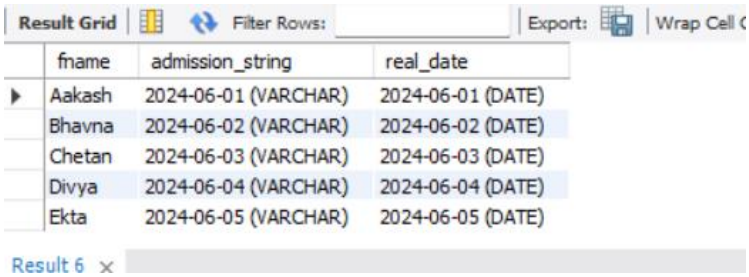
◆ 2. DATE / TIME Functions

✅ Query 1: Convert admission_date (string) → DATE

```
SELECT fname, admission_date, CAST(admission_date AS DATE) AS real_date
FROM students;
```

ANOTHER

```
SELECT
  fname,
  CONCAT(admission_date, ' (VARCHAR)') AS admission_string,
  CONCAT(CAST(admission_date AS DATE), ' (DATE)') AS real_date
FROM students;
```



🔼 Output:

fname	admission_date	real_date
Aakash	2024-06-01	2024-06-01
Bhavna	2024-06-02	2024-06-02
Chetan	2024-06-03	2024-06-03
Divya	2024-06-04	2024-06-04
Ekta	2024-06-05	2024-06-05

➔

✅ Query 2: Year and Month of Birth

```
SELECT fname, dob,
  YEAR(dob) AS birth_year,
  MONTH(dob) AS birth_month
FROM students;
```

🔼 Output:

fname	dob	birth_year	birth_month
Aakash	2003-05-20	2003	5
Bhavna	2004-03-15	2004	3
Chetan	2002-12-10	2002	12
Divya	2003-08-09	2003	8
Ekta	2005-01-01	2005	1

✅ Query 3: Days since Admission (Using DATEDIFF)

```
SELECT fname,
  DATEDIFF(CURDATE(), CAST(admission_date AS DATE)) AS days_passed
FROM students;
```

(Assume today's date is: `2025-06-19`)

🔼 Output:

fname	days_passed
Aakash	383
Bhavna	382
Chetan	381
Divya	380
Ekta	379

◆ 3. INTERVALS — Date me Add / Subtract

✅ Query 1: Add 1 Year in DOB

```
SELECT fname, dob,
  DATE_ADD(dob, INTERVAL 1 YEAR) AS dob_plus_1_year
FROM students;
```

📤 Output:

fname	dob	dob_plus_1_year
Aakash	2003-05-20	2004-05-20
Bhavna	2004-03-15	2005-03-15
Chetan	2002-12-10	2003-12-10

✅ Query 2: Subtract 2 Months in DOB

```
SELECT fname,
  DATE_SUB(dob, INTERVAL 2 MONTH) AS dob_minus_2mo
FROM students;
```

📤 Output:

fname	dob	dob_minus_2mo
Aakash	2003-05-20	2003-03-20
Bhavna	2004-03-15	2004-01-15

✅ Query 3: Age Calculation (TIMESTAMPDIFF)

```
SELECT fname,
  TIMESTAMPDIFF(YEAR, dob, CURDATE()) AS age
FROM students;
```

📤 Output:

fname	age
Aakash	22
Bhavna	21
Chetan	22
Divya	21
Ekta	20
