ORACLE SQL INTERVIEW CHEAT SHEET

(Functions, Differences, and Key Concepts)

🔹 1️⃣ STRING FUNCTIONS & COMPARISONS

Function	Description	Example	Interview Difference
`UPPER(str)`	Converts to uppercase	`UPPER('oracle') → ORACLE`	**UPPER vs INITCAP:** UPPER converts all, INITCAP capitalizes only first letter per word.
`LOWER(str)`	Converts to lowercase	`LOWER('SQL') → sql`	—
`INITCAP(str)`	Capitalizes first letter of each word	`INITCAP('oracle database') → Oracle Database`	—
`SUBSTR(str, start, len)`	Extract substring	`SUBSTR('ORACLE',2,3) → RAC`	**SUBSTR vs INSTR:** SUBSTR extracts; INSTR finds position.
`INSTR(str, sub)`	Finds position	`INSTR('DATABASE','A') → 2`	**INSTR vs LOCATE (MySQL):** Oracle only supports INSTR.
`LENGTH(str)`	Returns length	`LENGTH('DATA') → 4`	**LENGTH vs LENGTHB:** LENGTH gives characters; LENGTHB gives bytes.
`REPLACE(str, search, repl)`	Replace substring	`REPLACE('ABCD','B','Z') → AZCD`	**REPLACE vs TRANSLATE:**REPLACE substitutes substring;

			TRANSLATE replaces character by position.
`TRANSLATE(str, from, to)`	Character-wise translation	`TRANSLATE('12345','15','AB') → A234B`	—
`LTRIM/RTRIM/TRIM`	Remove leading/trailing chars	`TRIM('$' FROM '$$DATA$$') → DATA`	**TRIM vs LTRIM/RTRIM:** TRIM can do both sides, others one side only.
`LPAD/RPAD(str,len,pad)`	Pad string	`LPAD('45',5,'0') → 00045`	**LPAD vs RPAD:** left pad vs right pad.
`CONCAT(a,b)`	Combine strings	`CONCAT('HELLO','SQL') → HELLOSQL`	**CONCAT vs

🔹 2️⃣ NUMERIC FUNCTIONS

Function	Description	Example	Interview Difference
`ABS(n)`	Absolute value	`ABS(-5) → 5`	—
`CEIL(n)`	Next integer ≥ n	`CEIL(4.1) → 5`	**CEIL vs FLOOR:** CEIL rounds up, FLOOR rounds down.
`FLOOR(n)`	Largest integer ≤ n	`FLOOR(4.8) → 4`	—
`ROUND(n,dec)`	Round to decimals	`ROUND(12.456,2) → 12.46`	**ROUND vs TRUNC:** ROUND rounds up/down; TRUNC cuts off decimals.
`TRUNC(n,dec)`	Truncate decimals	`TRUNC(12.456,2) → 12.45`	—
`MOD(a,b)`	Remainder	`MOD(10,3) → 1`	**MOD vs REMAINDER:** MOD = m - n×FLOOR(m/n); REMAINDER uses nearest integer.
`POWER(a,b)`	a raised to b	`POWER(2,3) → 8`	—

`SIGN(n)`	Sign of number	`SIGN(-20) → -1`	—
`SQRT(n)`	Square root	`SQRT(25) → 5`	—

🔹 3️⃣ DATE & TIME FUNCTIONS

Function	Description	Example	Interview Difference
`SYSDATE`	Current date/time of DB	`SELECT SYSDATE FROM DUAL;`	**SYSDATE vs CURRENT_DATE:**SYSDATE = DB time zone; CURRENT_DATE = session TZ.
`CURRENT_DATE`	Current session date	—	—
`CURRENT_TIMESTAMP`	Timestamp with TZ	—	**SYSTIMESTAMP vs CURRENT_TIMESTAMP:**SYSTIMESTAMP = DB TZ, CURRENT_TIMESTAMP = session TZ.
`ADD_MONTHS(date,n)`	Add months	`ADD_MONTHS(SYSDATE,3)`	—
`MONTHS_BETWEEN(d1,d2)`	Month difference	`MONTHS_BETWEEN('12-DEC-24','12-JAN-25') → 1`	—
`NEXT_DAY(date,'DAY')`	Next weekday	`NEXT_DAY(SYSDATE,'MONDAY')`	—
`LAST_DAY(date)`	Last day of month	`LAST_DAY(SYSDATE)`	—
`ROUND(date,'fmt')`	Round to fmt	`ROUND(SYSDATE,'MONTH')`	**ROUND vs TRUNC (dates):** ROUND goes to nearest boundary, TRUNC just cuts.

`TRUNC(date,'fmt')`	Truncate date	`TRUNC(SYSDATE,'MONTH')`	—
`EXTRACT(part FROM date)`	Get year/month/day	`EXTRACT(YEAR FROM SYSDATE)`	—
`TO_CHAR(date, fmt)`	Date → String	`TO_CHAR(SYSDATE, 'DD-MON-YYYY')`	**TO_CHAR vs TO_DATE:** One converts to string, other back to date.
`TO_DATE(str,fmt)`	String → Date	—	—

🔹 4️⃣ CONVERSION FUNCTIONS

Function	Description	Example	Difference
`TO_CHAR(expr, fmt)`	Converts to string	`TO_CHAR(1234,'$9,999')`	—
`TO_NUMBER(expr, fmt)`	Converts to number	`TO_NUMBER('1234')`	—
`TO_DATE(str, fmt)`	Converts to date	`TO_DATE('12-11-25','DD-MM-YY')`	**TO_DATE vs CAST:** CAST is ANSI standard; TO_DATE is Oracle-specific.
`CAST(expr AS datatype)`	Type conversion	`CAST('123' AS NUMBER)`	**CAST vs CONVERT:** CONVERT is rarely used in Oracle; CAST is preferred.
`TO_TIMESTAMP(str, fmt)`	String → timestamp	—	**TO_TIMESTAMP vs TO_DATE:** TO_TIMESTAMP stores time fraction and TZ info.

🔹 5️⃣ NULL HANDLING & CONDITIONALS

Function	Description	Example	Interview Difference
`NVL(expr1,expr2)`	Replace NULL	`NVL(comm,0)`	**NVL vs NVL2:** NVL2 gives different results for NULL/non-NULL.
`NVL2(expr1,expr2,expr3)`	If expr1 not null → expr2 else expr3	—	—
`COALESCE(e1,e2,...)`	First non-null value	`COALESCE(comm,bonus,0)`	**COALESCE vs NVL:**COALESCE supports multiple arguments; NVL only 2.
`NULLIF(a,b)`	Returns NULL if equal	`NULLIF(sal,bonus)`	—
`DECODE(expr, val, result, default)`	IF-THEN logic	`DECODE(dept,10,'HR',20,'IT','Other')`	**DECODE vs CASE:**CASE is ANSI standard, more readable and supports conditions.
`CASE WHEN cond THEN val ELSE val END`	Conditional branching	`CASE WHEN sal>5000 THEN 'HIGH' ELSE 'LOW' END`	—

🔹 6️⃣ AGGREGATE FUNCTIONS

Function	Description	Example	Difference

`COUNT(*)`	Count rows	—	**COUNT(*) vs COUNT(col):** * counts all rows; col counts non-null.
`SUM(col)`	Total	—	—
`AVG(col)`	Average	—	—
`MIN/MAX(col)`	Smallest / largest	—	—
`LISTAGG(col, delimiter)`	Concatenate group values	`LISTAGG(name, ',') WITHIN GROUP (ORDER BY name)`	—
`STDDEV/VARIANCE(col)`	Stats functions	—	—

🔹 7️⃣ ANALYTIC / WINDOW FUNCTIONS

Function	Description	Example	Difference
`ROW_NUMBER()`	Sequential row number	`ROW_NUMBER() OVER (ORDER BY sal)`	**ROW_NUMBER vs RANK:** ROW_NUMBER no tie gaps; RANK creates gaps.
`RANK()`	Rank with gaps	`RANK() OVER (ORDER BY sal DESC)`	—
`DENSE_RANK()`	Rank without gaps	—	—
`LAG(expr [,offset,default])`	Previous row value	`LAG(sal) OVER (ORDER BY sal)`	**LAG vs LEAD:** LAG = previous row, LEAD = next row.
`LEAD(expr [,offset,default])`	Next row value	—	—

`FIRST_VALUE(expr)`	First value in window	`FIRST_VALUE(sal) OVER (...)`	—
`LAST_VALUE(expr)`	Last value in window	—	—
`NTILE(n)`	Divide rows into buckets	`NTILE(4) OVER (ORDER BY sal)`	—
`SUM(expr) OVER(...)`	Running total	—	—
`AVG(expr) OVER(...)`	Moving average	—	—

🔹 8️⃣ GENERAL / SYSTEM FUNCTIONS

Function	Description	Example	Difference
`USER`	Current DB user	`SELECT USER FROM DUAL;`	—
`UID`	User ID	`SELECT UID FROM DUAL;`	—
`SYSDATE`	DB current date	—	—
`SYSTIMESTAMP`	DB current timestamp	—	**SYSTIMESTAMP vs CURRENT_TIMESTAMP:** DB TZ vs Session TZ.
`DUMP(expr)`	Internal representation	—	—
`VSIZE(expr)`	Bytes used	—	—

`SYS_CONTEXT(namespace,param)`	Returns session env info	`SYS_CONTEXT('USERENV','SESSION_USER')`	—

🔹 9️⃣ SQL CLAUSE ORDER (Execution)

Step	Clause	Purpose
1	FROM	Source tables
2	ON	Join condition
3	WHERE	Filter rows
4	GROUP BY	Aggregate
5	HAVING	Group filter
6	SELECT	Projection
7	DISTINCT	Remove duplicates
8	ORDER BY	Sort
9	LIMIT/FETCH	Return subset

🔹 🔁 Common “Difference” Interview Questions

Topic	Difference
`DELETE vs TRUNCATE vs DROP`	DELETE = remove rows (can rollback); TRUNCATE = remove all rows (no rollback, faster); DROP = remove table structure.
`INNER JOIN vs OUTER JOIN`	INNER returns matching rows; OUTER includes unmatched rows too (LEFT/RIGHT/FULL).
`UNION vs UNION ALL`	UNION removes duplicates; UNION ALL keeps all.
`CHAR vs VARCHAR2`	CHAR fixed-length; VARCHAR2 variable-length.

`SYSDATE vs CURRENT_DATE`	SYSDATE = server timezone; CURRENT_DATE = session timezone.
`PRIMARY KEY vs UNIQUE`	Both enforce uniqueness, but PK = not null + unique, UNIQUE allows nulls.
`WHERE vs HAVING`	WHERE filters rows before GROUP BY; HAVING filters after aggregation.
`JOIN vs SUBQUERY`	JOIN merges multiple tables in one query; SUBQUERY executes nested query for filtering/aggregation.
`VIEW vs MATERIALIZED VIEW`	VIEW = virtual, no data; MVIEW = physical copy, refreshable.
`OLTP vs OLAP`	OLTP = transactional (insert/update/delete), OLAP = analytical (read-heavy, aggregation).

🧩 🔟 Useful Dual Table Examples

```
SELECT
  UPPER('oracle') AS upper_case,
  ROUND(123.456,2) AS rounded,
  TO_CHAR(SYSDATE,'DD-MON-YYYY HH24:MI:SS') AS today,
  NVL(NULL, 'default') AS nvl_example,
  DECODE(10, 10, 'TEN', 'OTHER') AS decode_test
FROM DUAL;
```
