Chapter 5: SELECT Statements

A **function** is a predefined block of code that accepts one or more arguments – values listed inside parentheses - and then returns a single value as output.

Simple Select clause: SELECT * FROM table_name

Alias: SELECT column "New name" FROM table_name; or SELECT column AS name_name FROM table_name;

Concatenation: SELECT col1 || col2 FROM table_name; or SELECT CONCAT (col1, col2) FROM table_name;

LTRIM/RTRIM: SELECT LTRIM (col, '-'), RTRIM(col, '-') FROM table_name;

TRIM: SELECT TRIM (LEADING '-' FROM col), TRIM (TRAILING '-' FROM col), TRIM ('-' FROM col) FROM table_name;

Distinct/Unique: SELECT DISTINCT col FROM table_name; (works)
SELECT col1, DISTINCT col2 FROM table_name; (doesn't work)

Dual table: SELECT 'hello' FROM dual; (displays hello)

INITCAP: SELECT INITCAP (col); (displays first letter in uppercase)

INSTR: SELECT INSTR (col, 'l', 1, 3); (displays index of third occurrence of 'l' from col (starts from 0))

SUBSTR: SELECT SUBSTR (col, -3, 2); (goes from end, back 3, select 2 letters)

Replace: SELECT REPLACE (col, 'word', 'non'); (replaces word with non)

LPAD/RPAD: SELECT RPAD (col, 20, '.') FROM; (adds . to the right until col reaches 20 chars)

TRUNC (chops, doesn't round), **ROUND** (rounds), **CEIL** (rounds up), **FLOOR** (rounds down)

Arithmetic: +-/*, MOD (col, 2), ABS (col), POWER (num, power), SQRT (num), GREATEST, LEAST

Date function: sysdate, ADD_MONTHS (col, 1), GREATEST, LEAST, LAST_DAY, MONTHS_BETWEEN, NEXT_DAY (col, 4), ROUND (TO_DATE('01-OCT-2011'), 'MONTH), TRUNC

NVL: SELECT NVL (col, 'hey'); (NULL replaced with 'hey') – SELECT NVL2(TO_CHAR (salary), 'poor', rich'); (poor when not null)

DECODE: SELECT DECODE (col, 'a', 'A', 'b', 'B', 'c', 'C', 'unknown');

SIGN: SELECT DECODE (SIGN (salary-40000), 0, 'good money', -1, 'Need more', 1, 'Donate');

CASE: SELECT salary, CASE WHEN salary<40000 THEN 'Need more' WHEN salary=40000 THEN 'Okay' ELSE 'Donate' END;
To Number: SELECT TO_NUMBER('2009') * 2;

Chapter 6: Restricting

Elements: (=, <, >, <=, >=, < >, !=, ^=), (Between, And, In, Or, Not, Like, Is Null), (And, Or, NOT)

Date: WHERE dob > TO_DATE ('02/11/1978', 'mm/dd/yyyy') has to be used when date not in OG format (11-FEB-1979)

Lower/Upper: SELECT upper(col1), lower(col2)

And/Between: WHERE salary > 30000 AND salary < 80000; WHERE salary BETWEEN 30000 AND 80000; (not include 30k, 80k)

Order of Precedence: Arithmetic first (+-/*), comparison second (=, <, >, <=, >=, < >, !=, ^=), logical third and last (NOT, AND, OR

Any/All: WHERE salary >ANY (30000, 40000); WHERE salary >ALL (30000, 40000);

Like: WHERE upper(city) LIKE '%D_';

Null: WHERE city IS NULL; WHERE city IS NOT NULL; (we can also use <> as null)

In/Not In: WHERE city IN ('Davis', 'Mavis'); WHERE city NOT IN ('Davis', 'Mavis');

Creating tables: CREATE TABLE table_name AS SELECT salary *2 new_salary FROM old_table; (only not NULL constraint copied)

Updating: UPDATE table_name SET col = '…' WHERE table_name_id = 1;

Chapter 7: Sorting

Order By: SELECT … FROM … ORDER BY col ASC;

NULLs: SELECT … FROM … ORDER BY col NULLS FIRST; NULLS LAST

Secondary Sorts: SELECT … FROM … ORDER BY col ASC, col 2 DESC;

Position: SELECT … FROM … ORDER BY 1, 2 DESC;

Column vs Alias: SELECT NVL (salary, 0) pay FROM patient ORDER BY pay;

Chapter 8: Group By

SUM: SELECT fname, city, SUM (salary) FROM patient GROUP BY fname, city;

DISTINCT: SELECT city, SUM (DISTINCT salary) FROM patient GROUP BY city;

AVG: SELECT city, AVG (nvl (salary, 0)) FROM patient GROUP BY city HAVING AVG (salary) > 20000 ORDER BY 1;

COUNT: SELECT city, COUNT (*) FROM patient GROUP BY city HAVING COUNT (*) > 1;

MAX: SELECT fname, city, COUNT (*), AVG (salary), MAX (salary) FROM patient GROUP BY fname, city HAVING COUNT (*) > 2;

MIN: SELECT city, MIN (salary) FROM patient WHERE city is NOT NULL GROUP BY city HAVING COUNT (*) > 1 ORDER BY 1 DESC;

Dates: Can use (min, max, count), Cannot use (avg, sum)

Chapter 9: Subqueries

Using Subqueries – Multiple Rows: SELECT fname, lname FROM patient WHERE **patient_id** IN (
SELECT **patient_id** FROM patient_disease WHERE **disease_id** = (
SELECT **disease_id** FROM disease WHERE disease_desc = 'Cancer'));

More examples: SELECT disease_desc FROM disease WHERE disease_id IN (
SELECT disease_id FROM patient_disease WHERE patient_id = (
SELECT patient_id FROM patient WHERE fname = 'john' AND lname 'Doe'));

Multiple Column Subquery: SELECT patient_id FROM patient WHERE (fname, lname) IN (
SELECT fname, lname FROM special_names);

Group functions and subqueries: SELECT fname, lname, dob FROM patient WHERE MONTHS_BETWEEN (sysdate, dob) > (
SELECT AVG (MONTHS_BETWEEN (sysdate, dob)) FROM patient);

More examples: SELECT fname, lname, salary FROM patient WHERE salary > (
SELECT AVG (salary) FROM patient);

Create table: CREATE TABLE new_table2 AS SELECT patient_id, salary * 2 Increase FROM patient WHERE (fname, lname) IN (
SELECT fname, lname FROM special_names;

Update Subqueries: UPDATE patient SET salary = salary * 2 WHERE patient_id IN (
SELECT patient_id FROM patient_disease WHERE disease_id = (
SELECT disease_id FROM disease WHERE disease_desc = 'Cancer'));

Delete Subqueries: DELETE FROM patient_disease WHERE disease_id IN (
SELECT disease_id FROM disease WHERE disease_desc = 'Cancer');