Creating/Altering Tables

CREATE TABLE <TableName> (
    <columnName> <columnType>, …,
    CONSTRAINT <constraintName> <constraintType> (<columnName(s)>), ….);
Column Types: Number(#), varchar2(#), date
Adding via Alter: ALTER TABLE <TableName> ADD <valid column or constraint>;

Constraint Examples

CONSTRAINT Table_PK PRIMARY KEY (keyColumn);
CONSTRAINT Table_FKCol_FK FOREIGN KEY (FKCol) REFERENCES OtherTable (key);
CONSTRAINT Table_UQ UNIQUE (uniqueColOne, uniqueColTwo);
CONSTRAINT Table_SetInVal CHECK (Set IN ('val1', 'val2'));
CONSTRAINT Table_ValNotNull CHECK (Val IS NOT NULL);
    Also: ALTER TABLE <TableName> MODIFY (<colName> NOT NULL);
CONSTRAINT Table_StringLikeVal CHECK (StringCol LIKE 'pattern');
CONSTRAINT Table_RangeVal CHECK (Range < upperLimit AND Range > lowerLimit);

Nulls

- 'Null' means UNKNOWN → causes undefined behavior in some operations
- Null groups into its own group
- Ignored in all aggregate functions except COUNT(*)
- NVL(exp1, exp2) = exp1 if exp1 is not null, else exp2

DML

INSERT INTO <TableName> [{<columnName>}] VALUES ({<columnValues>});
DELETE FROM <TableName> WHERE <boolExp>;

UPDATE <TableName> SET <columnName> = <valueExpression> [WHERE <boolExp>];
Update on a condition:
    UPDATE Table SET col = 100 WHERE condition = 'yes';
Update from another table:
    UPDATE ATable A SET col = (
        SELECT B.value FROM BTABLE B WHERE A.id = B.id)
    WHERE id IN (SELECT B.id FROM BTABLE B);

SELECT [DISTINCT] _ FROM _ [WHERE _] [GROUP BY _ [HAVING _]] [ORDER BY _]
    Execution Order: FROM/WHERE/GROUP BY/HAVING/ORDER BY/SELECT
Aggregation Operators: COUNT(*), COUNT(x), MIN(x), MAX(x), SUM(x), AVG(x)
    Non-Aggregated columns in the SELECT clause must be in the GROUP BY clause
Renaming in the SELECT Clause: <colName or aggregation or expression> AS <newName>
    Formatting Numbers: to_char(<number>, 990.99)
    String concatenation: 'string1' || 'string2'