SQL => è un linguaggio con varie funzionalità, contiene sia il DDL (Data Definition Language) che il DML (Data Manipulation Language), ne esistono varie versioni e dal 1983 è diventato uno standard

CREATE TABLE => definisce uno schema di relazione e ne crea un'istanza vuota in cui specifica ATTRIBUTI, DOMINI e VINCOLI (CREATE TABLE Impiegato (Matricola (ATTRIBUTO) CHAR(6) (DOMINIO) PRIMARY KEY (VINCOLO), Nome CHAR(20) NOT NULL, Stipendio NUMERIC(9) DEFAULT 0 ...)
(implica il NOT NULL)

DOMINIO => i domini si dividono in: ELEMENTARI o PREDEFINITI (CARATTERE [carattere o stringhe], NUMERICI [esatti e approssimati], DATA, ORA, INTERVALLI DI TEMPO, BOOLEANI o BLOB/CLOB [binary/character long object ⇒ per grandi immagini e testi]) e DEFINITI DALL'UTENTE (semplici ma vincolanti, l'istruzione è CREATE DOMAIN [CREATE DOMAIN Voto AS SMALLINT DEFAULT NULL CHECK (value >= 18 AND value <= 30)])

VINCOLI INTRARELAZIONALI => NOT NULL, UNIQUE (definisce chiavi), PRIMARY KEY (se forma da solo la chiave), CHECK

VINCOLI INTERRELAZIONALI => CHECK, REFERENCES (crea un legame tra i valori di un attributo di una tabella relativo a quello di un attributo di una tabella master esterna [ATTRIBUTO(S) dominio REFERENCES tabella master (ATTRIBUTO(S))]), FOREIGN KEY (crea un legame tra i valori di più attributi di una tabella relativo a quello di più attributi di una tabella master esterna [* va indicato dopo la definizione degli attributi !. FOREIGN KEY (ATTRIBUTI(S)) REFERENCES ATTRIBUTI (N)])

MODIFICHE DEGLI SCHEMI => ALTER (DOMAIN, TABLE ...), DROP (DOMAIN, TABLE ...)

DEFINIZIONE DEGLI INDICI => è rilevante per le prestazioni ma è a livello fisico e non logico; in passato era importante perché era l'unico modo per definire le chiavi, CREATE INDEX

OPERAZIONI SUI DATI => SELECT (interrogazione [LISTA ATTRIBUTI]); FROM (interrogazione [LISTA TABELLE]); WHERE (interrogazione [CONDIZIONE]); INSERT (+ INTO modifica [INSERISCE]); DELETE (+ FROM modifica [CANCELLA]); UPDATE (modifica [ASSEGNAMENTO (INCREMENTARE O)]); ABBREVIAZIONI (esempio => SELECT P.Nome AS N, P.Reddito AS R, FROM Persone AS P, WHERE P.Età < 30)

CONDIZIONE "LIKE" => utilizzata per configurare una condizione nella ricerca es. WHERE Nome like "A_d%" (JOLLY ?! - [qualsiasi carattere singolo]; %/% [zero o più caratteri]; # [cifra singola]; [elencocaratteri] viene preso qualsiasi carattere incluso; [!elencocaratteri] vengono esclusi i caratteri elencati; [a-zA-Z0-9] vengono presi tutti i caratteri all'alfanumerici !* l'intervallo è valido solo se espresso in ordine crescente)

AND o OR => applicati ad una condizione per congiungere un'altra

DISTINCT => usato per selezionare i valori diversi in una occorrenza o relazione di occorrenze (SELECT DISTINCT, Cognome, Nome)

FROM e JOIN => FROM usato se si ricerca in una sola relazione, il DBMS utilizza JOIN se ci sono più relazioni nella ricerca (o prodotti cartesiani) (il JOIN può essere esplicitato come comando o meno ed avere condizione esplicita con ON [FROM Tab1 JOIN Tab2 ON Cond Join])
* la capacità dei DBMS di ottimizzare le interrogazioni rende non necessario preoccuparsi dell'efficienza quando si specifica un'interrogazione, è invece più importante preoccuparsi della chiarezza

JOIN => JOIN collega le due tabelle; NATURAL JOIN collega le tabelle con la loro relazione comune; OUTER JOIN collega le due tabelle mediante la condizione espressa con l'ON (dichiarare quale è il primo di sinistra (LEFT, RIGHT, FULL)

ORDER BY => ordina i risultati mediante la condizione/attributo esplicitato (ORDER BY Nome)

UNION => consente di unire i risultati e i duplicati vengono eliminati anche dalle proiezioni a meno che non si usi l' ALL (si fa solo a livello esterno e non nidificato)

EXCEPT => consente di effettuare una differenza (si può esprimere anche con i SELECT nidificati)

INTERSECT => consente di fare un'intersezione

INTERROGAZIONI NIDIFICATE => permettono il confronto tra un attributo e il risultato di una sotto-interrogazione (SELECT Nome, Reddito FROM Persone WHERE Nome = → (SELECT Padre FROM Paternità WHERE Figlio = "Franco")

ANY, IN, ALL => IN e NOT IN (spesso meno usati per costruire intersezioni e differenze di tabelle); ANY (usato nella clausola WHERE, è un confronto che consente uno se è vero perlomeno uno dei valori dell'elenco; ALL (vero se il confronto è vero per tutti gli elementi)

EXIST => controlla se vengono restituite righe dall'esecuzione della sottoquery, è vera se la SELECT nidificata produce righe come risultato (NOT EXIST il contrario); QUANTIFICAZIONE ESISTENZIALE

OPERATORI AGGREGATI => COUNT (conteggio occorrenze SELECT count(*) as ...) AVG (average ci dà una media delle occorrenze SELECT avg(ATTRIBUTO) as ...); MIN (restituisce le occorrenze con es. il reddito minimo WHERE Reddito = (SELECT min(Reddito) FROM Persone) MAX (ci restituisce le occorrenze con es. il reddito massimo es. WHERE Reddito = (SELECT max(Reddito) FROM Persone)

OPERATORI AGGREGATI E RAGGRUPPAMENTI => GROUP BY (raggruppa le occorrenze uguali); HAVING (condizione che indica di cosa una determinata caratteristica)

INSERIMENTO => l'ordinamento degli attributi e dei valori è significativo, le due liste devono avere lo stesso numero di elementi; se la lista di attributi è omessa si fa riferimento a tutti gli attributi della relazione secondo l'ordine in cui sono stati definiti, se la lista degli attributi non contiene tutti gli attributi della relazione per gli altri viene inserito NULL o un valore di default