SELECT

1. SELECT * FROM tbl
   WHERE condition
   ORDER BY c1 ASC, c2 DESC
2. SELECT DISTINCT c1, c2 FROM tbl
3. SELECT c1, AGG(expr) AS c2
   FROM tbl
   GROUP BY c1
   HAVING c2 > v
4. SELECT c1 || ' ' || c2 AS "ABC", FROM tbl

KEYS

1. Primary – Entity Integrity Constraint
- unique
- cannot be NULL
2. Foreign – Referential Identity Entity
- value is from PK of some other tables
- can be NULL

EXPR

1. c1 = 'string'
2. IS NULL / IS NOT NULL
3. LIKE '%A_'
4. c1 = 1 OR c1 = 2 >> c1 IN (1, 2)
5. c1 = 1 AND c1 = 2 >> c1 BETWEEN 1 AND 2 ** inclusive **

DATA (DML)

1. INSERT INTO tbl (c1, c2, ...)
   VALUES (v1, 'v2', ...)

1.2 INSERT ALL
   INTO tbl VALUES (v1, 'v2'...),
   INTO tbl VALUES (v1, 'v2'...)
   SELECT * FROM dual;

2. INSERT INTO tbl (c1, c2, ...)
   SELECT c1, c2... FROM tbl2
   WHERE conditions
3. UPDATE tbl
   SET c1 = v1, c2 = v2...
   WHERE conditions
4. DELETE FROM tbl
   WHERE conditions

TABLE (DDL)

1.1 CREATE TABLE tbl (
   c1 TYPE(length) NULL / NOT NULL,
   c2 TYPE(length) constraint
   CONSTRAINT con_name
    PRIMARY KEY (c1, c2...),
   CONSTRAINT con_name
    FORIEGN KEY (c1, c2...) REFERENCES tbl_parent (PK_c1, PK_c2...),
   CONSTRAINT con_name
    UNIQUE (c1, c2...),
   CONSTRAINT con_name
    CHECK (validation >>c1 BETWEEN 18 AND 40)
   ON DELETE [CASCADE / SET NULL]
);

1.2 CREATE TABLE tbl_new AS
   SELECT * FROM tbl_old;

2. DROP TABLE tbl;
3. ALTER TABLE tbl
   ADD COLUMN c1 TYPE(length) constratint
   ADD CONSTRAINT con_name FK ...
4. ALTER TABLE tbl
   DROP COLUMN c1
   DROP CONSTRAINT con_name FK ...
5. ALTER TABLE tbl
   MODIFY (c1 UNIQUE);
6. ALTER TABLE tbl
   RENAME TO tbl_new
   RENAME COLUMN c1 TO c1_new
   OR CHANGE c1 c1_new TYPE

** multiple ADD / MODIFY col using () and separate field with ,
** multiple DROP col >> DROP col1, DROP c2...

TYPE

NUMBER, DECIMAL(length), CHARACTER(), VARCHAR(), DATE **do not specify length 'YYYY-MM-DD', TIME

CONSTRAINT

FK: relational integrity

JOIN

G syntax: SELECT * FROM tblA A
   ... JOIN tblB B
    ON join-codition

1. FULL OUTER JOIN (A U B) >> n*n
2. (A U B) – (A in B)
   WHERE A.id IS NULL
      OR B.id IS NULL
3. INNER JOIN (A in B)
4. LEFT JOIN (A)
5. (A) – (A in B)
   WHERE tblB.id IS NULL

EXTRA

1. CREATE VIEW view_name AS SELECT...
2. DROP VIEW view_name

COMMIT

1. SET AUTOCOMMIT ON / OFF;
2. ROLLBACK;
3. COMMIT;

TRANSACTION

- typically consists of one or more database Manipulation operations like inserting, updating or deleting data from tables.
   SET TRANSACTION READ WRITE;
- main purpose is to maintain the ACID properties
A(tomicity) C(onsistency) I(solation) D(urability)

Return a column

```
SELECT productcode, productname, quantityinstock
FROM dbs211_products
WHERE productname LIKE 'co%' OR productname LIKE 'Co%'
OR productname LIKE '%Co%' OR productname LIKE '%co%'
ORDER BY quantityinstock;
```

```
SELECT ProductCode, ProductName, BuyPrice, MSRP, (MSRP - BuyPrice) AS markup,
    ROUND((100 * (MSRP - BuyPrice) / BuyPrice), 1) AS percmarkup
FROM dbs211_products
WHERE ROUND((100 * (MSRP - BuyPrice) / BuyPrice), 1) > 140.0
ORDER BY percmarkup;
```

```
CREATE TABLE Orders      (
    order_id        NUMBER,
    customer_id     NUMBER,
    order_date      DATE,
    total_amount    NUMBER,
    status          VARCHAR2    (50),
    Constraint FK_customer_id
    FOREIGN KEY     (customer_id)   REFERENCES Customer(customer_id))
```