DDL(Data Definition Language): Schema

* CREATE/ALTER/DELETE TABLE & TABLES’ Attributes

DML(Data Manipulation Language): Actual Data, Instance

* INSERT/DELETE/MODIFY tuples in TABLES

Relation or Table: multiset of tuples. Duplicates are allowed. No order.

Attribute: or Column: typed data entry present in each tuple in the relation. Atomic type (Single value)

Row or tuples or record: single entry in the table having attributes specified by schema, Not NULL

Number of tuples: cardinality

Number of columns: arity

Atomic types:

* Characters: CHAR(), VARCHAR()
  + Size of VARCHAR = size of data + 1 byte(length of string)
* Numbers: INT, BIGINT, SMALLINT, FLOAT
* Others: MONEY, DATE

SFW

* SELECT
  + SELECT a\*2 from TABLE -> arithmetic operation
  + SELECT a as A\_RESULT from TABLE -> Alias
  + SELECT a || ’, ’ || b as A\_AND\_B from TABLE -> Concatenate
  + SELECT DISTINCT Category -> remove redundancy
* FROM
* WHERE
  + Condition on individual tuples
  + WHERE Pname LIKE ‘%Gizmo%’ -> pattern matching on strings using ‘%’ for any sequence of characters & ‘\_’ single character
  + WHERE a BETWEEN 10 AND 30
* ORDER BY ->default: ASC
  + ORDER BY Price, Pname DESC
* GROUP BY
* HAVING
  + Condition on aggregates

FROM & WHERE -> GROUP BY -> SELECT

Selection -> ROWS, WHERE clause

Projection -> 표를 만든다! COLUMNS, SELECT clause

JOIN OPERATION

* using foreign key

SELECT Pname, Price

FROM Product, Company

WHERE Manufacturer = CName AND Country=’Japan’ AND Price <= 200;

* using JOIN ~ ON

SELECT Pname, Price

FROM Product, Company

JOIN Company ON Manufacturer = Cname AND Country = ‘Japan’

WHERE Price <= 200;

MULTISET OPERATOR

* INTERSECT
  + SELECT R.A

FROM R, S

WHERE R.A = S.A

INTERSECT

SELECT R.A

FROM R,T

WHERE R.A=T.A;

* + Number of attributes in SELECT clause should be the same
  + Type of attributes in SELECT clause should be the same
* UNION
  + SELECT R.A

FROM R, S

WHERE R.A = S.A

UNION

SELECT R.A

FROM R,T

WHERE R.A=T.A;

* + No Duplicates
* UNION ALL
  + SELECT R.A

FROM R, S

WHERE R.A = S.A

UNION ALL

SELECT R.A

FROM R,T

WHERE R.A=T.A;

* + All rows including duplicates
* MINUS
  + SELECT R.A

FROM R, S

WHERE R.A = S.A

MINUS

SELECT R.A

FROM R,T

WHERE R.A=T.A;

* + No intersection between two sets -> return 0

ALL / ANY / EXISTS

Aggregation -> SUM, COUNT, MIN, MAX, AVG in SELECT clause

Except COUNT, all aggregation applies to single attributes

* COUNT -> include duplicates until DISTINCT applied

NULL

* SELECT \*

FROM person

WHERE age < 25 or age >= 25 or age IS NULL

* IS NULL / IS NOT NULL

Outer Join

* LEFT OUTER JOIN ~ ON-> include left tuple even there is no match
* RIGHT OUTER JOIN ~ ON-> include right tuple even there is no match
* FULL OUTER JOIN ~ ON-> include both left and right tuple even there is no match

INSERT INTO table\_name (columns ~~)

VALUES (values ~~);

* Insert New Records

INSERT INTO table2

SELECT \* FROM table1

WHERE condition;

* Insert copied records from other table (all columns)

INSERT INTO table2(columns, columns, columns)

SELECT columns, columns, columns FROM table1

WHERE condition;

* Insert copied records from other table (some columns)

INSERT ALL

INTO table (columns~) values (values~)

INTO table (columns~) values (values~)

INTO table (columns~) values (values~)

SELECT \* FROM dual;

=

INSERT INTO table (columns~) value (values ~);

INSERT INTO table (columns~) value (values ~);

INSERT INTO table (columns~) value (values ~);

UPDATE

* UPDATE table

SET column1 = value1, column2 = value2, column3 = value3, …

WHERE condition;

DELETE

* DELETE FROM table WHERE condition;

CREATE TABLE table (

column1 datatype,

column2 datatype,

column3 datatype,

column4 datatype NOT NULL,

……

);

* Create New Table

CREATE TABLE new\_table\_name AS

SELECT column1, column2

FROM existing\_table\_name

WHERE ……;

* Create table using other table

DROP TABLE table\_name;

* drop the existing table

TRUNCATE TABLE table\_name;

* delete data, maintain table schema

ALTER TABLE table\_name

ADD table\_column datatype;

* add, delete, or modify columns in existing table

ALTER TABLE table\_name

DROP COLUMN column\_name;

ALTER TABLE table\_name

MODIFY column\_name datatype;

* Only for no value column

ALTER TABLE table\_name

RENAME COLUMN old\_name TO new\_name;

* rename column

ALTER TABLE table\_name

RENAME TO new\_name;

* rename table

SELECT TOP

* SELECT column names

FROM table\_name

WHERE ROWNUM <= number;

View

* Create view v as <query expression>