UNIT- III

- **Structured Query Language**: Basic SQL Querying Using Select and Where clauses, Arithmetic & Logical Operations, SQL Functions (Date and Time, Numeric to String Conversion). Creating Tables with Relationship, Implementation of Key and Other Integrity Constraints, Set Operations, Nested Queries, Sub Queries, Grouping, Aggregation, Ordering, Implementation of Different Types of Joins, Views (Updatable and Non-Updatable).

Structured Query Language – The Basics

BASIC SQL

SQL stands for Structured Query Language

SQL is a standard language for accessing and manipulating databases.

SQL became a standard of the

- American National Standards Institute (ANSI) in 1986

- International Organization for Standardization (ISO) in 1987

What Can SQL do?

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views

SQL Environment

- **Catalog**
  - A set of schemas that constitute the description of a database
- **Schema**
  - The structure that contains descriptions of objects created by a user (base tables, views, constraints)
- **Data Definition Language (DDL)**
  - Commands that define a database, including creating, altering, and dropping tables and establishing constraints
- **Data Manipulation Language (DML)**
  - Commands that maintain and query a database
- **Data Control Language (DCL)**
  - Commands that control a database, including administering privileges and committing data

Overview of SQL

- Data Definition Language
    - Creating tables
- Data Manipulation Language
    - Inserting/Updating/Deleting data
    - Retrieving data
        - Single table queries
        - Where
        - Joins
        - Grouping

SQL

- SQL is a data manipulation language.

- SQL is not a programming language.

- SQL commands are interpreted by the DBMS engine.

- SQL commands can be used interactively as a query language within the DBMS.

- SQL commands can be embedded within programming languages.

3 Types of SQL Commands

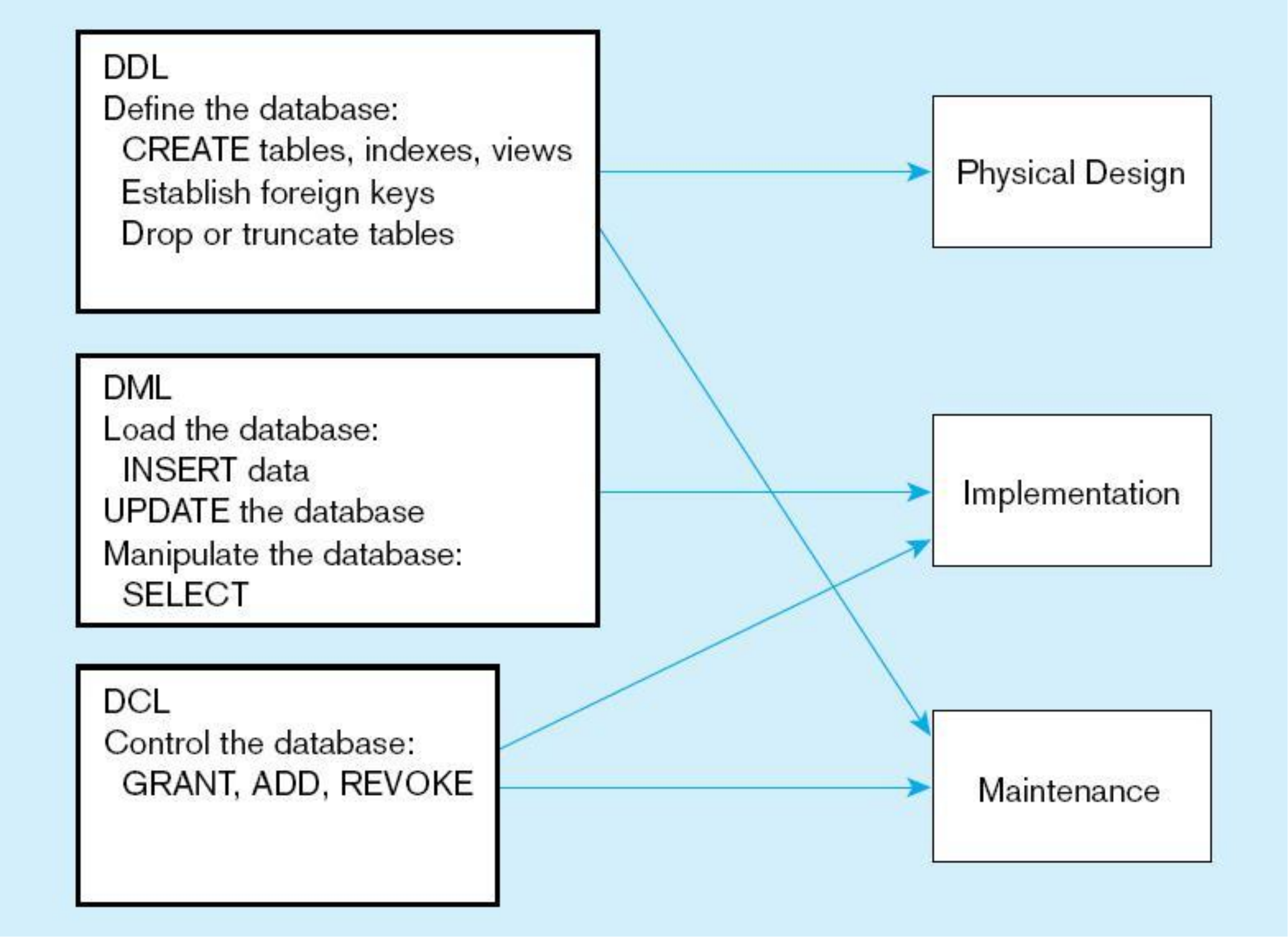
- **Data Definition Language (DDL):**
  - Commands that define a database - Create, Alter, Drop
- **Data Manipulation Language (DML)**
  - Commands that maintain and query a database.
- **Data Control Language (DCL)**
  - Commands that control a database, including administering privileges and committing data.

Data Manipulation Language (DML)

Four basic commands:

- INSERT
- UPDATE
- DELETE
- SELECT

DDL, DML, DCL, and the database development process



DDL
Define the database:
CREATE tables, indexes, views
Establish foreign keys
Drop or truncate tables

DML
Load the database:
INSERT data
UPDATE the database
Manipulate the database:
SELECT

DCL
Control the database:
GRANT, ADD, REVOKE

Physical Design

Implementation

Maintenance

Tables in SQL

Table name

Attribute names

Product

PName	Price	Category	Manufacturer
Gizmo	19.99	Gadgets	GizmoWorks
Powergizmo	29.99	Gadgets	GizmoWorks
SingleTouch	149.99	Photography	Canon
MultiTouch	203.99	Household	Hitachi

Tuples or rows

Tables Explained

- The *schema* of a table is the table name and its attributes:

Product(PName, Price, Category, Manfacturer)

- A *key* is an attribute whose values are unique;

Product(<u>PName</u>, Price, Category, Manfacturer)

Steps in Table Creation

1. Identify data types for attributes
2. Identify columns that can and cannot be null
3. Identify columns that must be unique (Primary key)
4. Identify primary key–foreign key mates
5. Determine default values
6. Identify constraints on columns (domain specifications)
7. Create the table and associated indexes

SQL Query

Basic form:

```
SELECT  <attributes>
FROM     <one or more relations>
WHERE  <conditions>
```

Notation

Input Schema

Product(PName, Price, Category, Manfacturer)

```
SELECT   PName, Price, Manufacturer
FROM      Product
WHERE   Price > 100
```

Answer(PName, Price, Manfacturer)

Output Schema

- Case insensitive:
  - Same: SELECT Select select
  - Same: Product   product
  - Different: ‘Seattle’ ‘seattle’

- Constants:
  - ‘abc’ - yes
  - “abc” - no

SQL DATA TYPES

- Character (n) or varchar
  - This data type represents a fixed length string of exactly ‘n’ characters where ‘n’ is greater than zero and should be an integer.

Example

```
name character(10);
name varchar(10)
```

SQL DATA TYPES cont…

- Varchar2(n)

This data type represents a varying length string whose maximum length is ‘n’ characters.

Example

```
    name varchar2(n);
```

- number(p,q)

  syntax

  ```
  number(l,d)
  ```

  Stores numeric data, where ‘l’ stand for length and ‘d’ for the number of decimal digits.

Example

```
price number (6,2);
```

- Integer

  An integer represents a signed integer represents a signed integer decimal or binary.

Example

```
Roll_no integer(5);
```

- Float (n)

  A floating point number, with precision of at least n digits.

Example

```
Rate float(5,2) ;
```

- Date

  A calender data containing a (four_digital year, month and day of the month).

Example

```
Date_of_birth date ;
```

- Time

  The time of day, in hours, minutes and seconds.

Example

Arrival_time time;

Data Definition Language

- It is used to create a table, alter the structure of a table and also drop the table.
- Create command
- Alter command
- Drop command
- Truncate command

Create Command

- It is used to create a table.

Syntax

```
Create table <table name>
( columnname1 datatype1,
columnname2 datatype2,  etc…);
```

Example

```
SQL> create table emp
(empno number(4),
ename varchar2(30),
salary  number(10,2),
deptno number(2));
```

Alter Command

- It is used to add a new column or modify existing column definitions.

Syntax

```
Alter table <table name>
Add ( new columnname1 datatype1,
newcolumnname2 datatype2,  etc...);

Alter table <table name>
Modify ( oldcolumnname1 datatype1,
oldewcolumnname2 datatype2,  etc...);
```

Example

```
SQL> alter table emp
add(comm number(6,2));
```

```
SQL> alter table emp
modify(empno number(5));
```

Removing Tables

- DROP TABLE statement allows you to remove tables from your schema:
  - DROP TABLE CUSTOMER_T

Drop Command

- This command is used to delete a table. [delete the contents (records and structure)].

Syntax

```
Drop table <table name>;
```

Example

```
SQL> drop table emp;
```

Truncate Command

- This command is used to delete the records but retain the structure.

Syntax

```
Truncate  table <table name>;
```

Example

```
SQL> truncate table emp;
```

Rename a table

Syntax

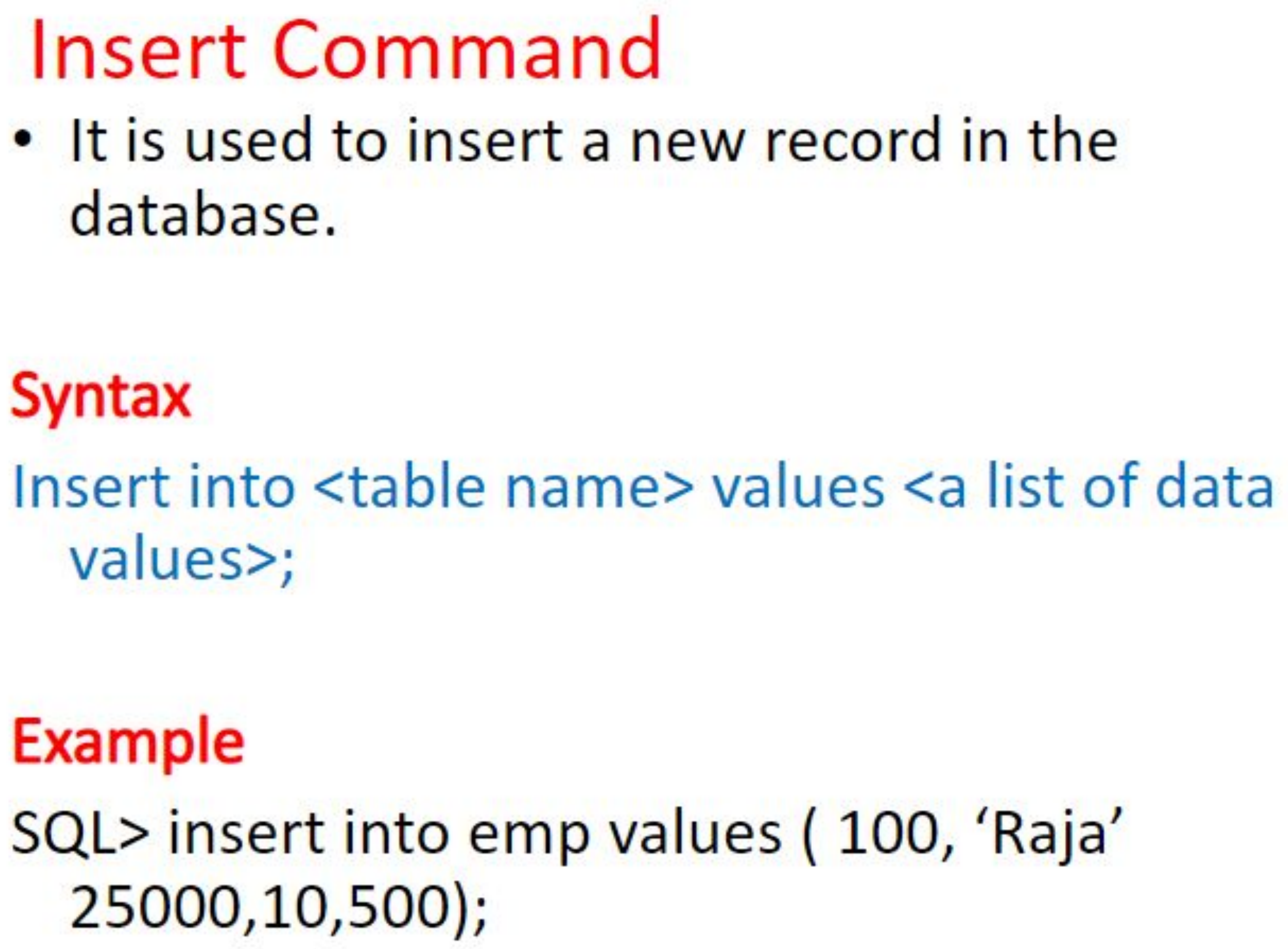
```
Rename <oldtablename> to <newtablename>;
```

Example

```
SQL> Rename emp to employee;
```

Data Manipulation Language

Insert, Update, Delete



Insert Command

- It is used to insert a new record in the database.

Syntax

Insert into <table name> values <a list of data values>;

Example

```
SQL> insert into emp values ( 100, ‘Raja’
   25000,10,500);
```

```
INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)
VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');
```

```sql
create table Info(id integer, Cost integer, city varchar(200));
insert into Info(id, Cost,city) values(1, 100,"Pune");
insert into Info(id, Cost,city) values(2, 50, "Satara");
insert into Info(id, Cost,city) values(3, 65,"Pune");
insert into Info(id, Cost,city) values(4, 97,"Mumbai");
insert into Info(id, Cost,city) values(5, 12,"USA");
select * from Info;
```

Inserting Data into a Table

INSERT INTO tablename (column-list) **VALUES (value-list)**

PUTS ONE ROW INTO A TABLE

```
INSERT INTO COURSE
  (COURSE_CODE, COURSE_NAME, CREDIT_HOURS)
VALUES (‘MIS499’,’ADVANCED ORACLE’,4);
```

More on Inserting Data

```
INSERT INTO COURSE
VALUES ('MIS499','ADVANCED ORACLE',4);
```

COLUMN LIST IS OPTIONAL IF YOU PLAN TO INSERT A VALUE IN EVERY COLUMN AND IN THE SAME ORDER AS IN THE TABLE

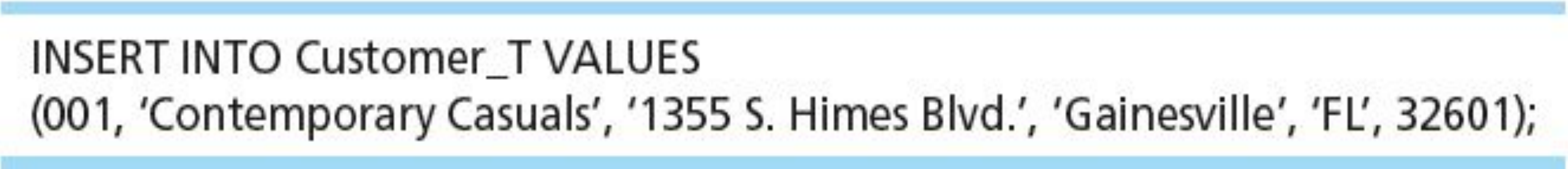
```
INSERT INTO COURSE
(COURSE_NAME, COURSE_CODE,  CREDIT_HOURS)
VALUES ('ADVANCED ORACLE','MIS499',4);
```

COLUMN LIST IS NEEDED TO CHANGE THEORDER
- MUST MATCH VALUE LIST

NOTE - TABLE STILL HAS THE ORIGINAL COLUMN ORDER

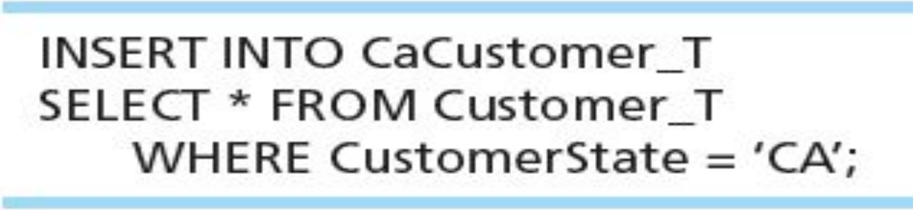
Insert Statement

- Adds one or more rows to a table
- Inserting into a table



```
INSERT INTO Customer_T VALUES
(001, 'Contemporary Casuals', '1355 S. Himes Blvd.', 'Gainesville', 'FL', 32601);
```

- Inserting from another table



```
INSERT INTO CaCustomer_T
SELECT * FROM Customer_T
     WHERE CustomerState = 'CA';
```

update Command

- Changes can be made by using update command.

Syntax

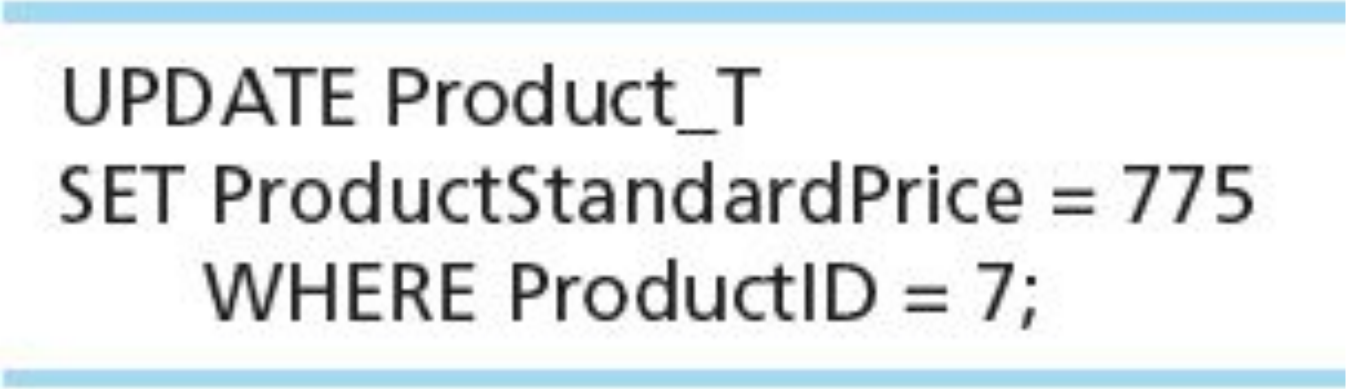
```
Update <table name> set filedname = values
  where <Condition>;
```

Example

```
SQL> update emp set comm=500 where
  eno=100;
```

Update Statement

- Modifies data in existing rows



```
UPDATE Product_T
SET ProductStandardPrice = 775
     WHERE ProductID = 7;
```

Updating Data

```
UPDATE COURSE SET HOURS=5;
```

CHANGES EVERY ROW

```
UPDATE COURSE SET HOURS=5
WHERE COURSE_CODE=‘MIS220’
```

CHANGES ONE ROW

```
UPDATE COURSE SET HOURS=3
WHERE COURSE_CODE LIKE ‘MIS%’
```

CHANGES A GROUP OF ROWS

Updating and Integrity Constraints

You Can Change The Value of a Foreign Key as long as The New Value also complies with Referential Integrity Constraints.

Primary Key values can be updated as long as there are No Rows in other Tables with Foreign Keys with the same value

DOES NOT MATTER IF CONSTRAINT IS RESTRICTED OR CASCADED

Integrity Error

```
SQL> UPDATE COURSE
    SET COURSE_CODE='MIS221‘
    WHERE COURSE_CODE='MIS220';
UPDATE COURSE


ERROR :
integrity constraint violated - child record found
```

Example

```
SQL>SQL> update emp set comm=1000;
SQL> update emp set comm=comm+500;
SQL> update emp set ename =‘Raj  kumar’ where
   ename =‘Raj’;
 SQL> update emp set comm=2000 where ename
   =‘Raja’ and salary>=30000; SQL> update emp set
   comm =40 where comm is null;
SQL> update emp set salary = salary*0.1 where
   comm is notnull;
SQL> update emp set salary =50000, comm=1500
   where eno=27;
```

Delete Command

- Rows can be deleted using delete command.

Syntax

```
Delete from <table name> where <Condition>;
```

Example

```
SQL> delete from emp where ename =‘abc’;
 SQL> select * from emp;
```

Example

```
SQL> delete from emp where salary<30000;
 SQL> delete from emp;
```

Delete Statement

Removes rows from a table

Delete certain rows

```
DELETE FROM CUSTOMER_T WHERE
  CUSTOMERSTATE = ‘HI’;
```

Delete all rows

```
DELETE FROM CUSTOMER_T;
```

Deleting Data

*Be careful!! This deletes ALL of the rows in your table. If you use this command in error, you can use **ROLLBACK** to undo the changes.*

DELETE COURSE;

Deletes All Rows

DELETE COURSE **WHERE** COURSE_CODE = ‘MIS220’;

Deletes Specific Rows

DELETE COURSE **WHERE** HOURS=4;

Deletes A Group Of Rows

DELETE COURSE **WHERE** HOURS<4;

Deleting and Integrity Constraints

```
SQL> DELETE COURSE
   WHERE COURSE_CODE='MIS220';


ERROR at line 1:
integrity constraint  violated - child record
found
```

SELECT STATEMENTS

- The select command is used to retrieve data from an oracle database.

Syntax

Select <field names> from <table name> where <condition>

Example

SQL> select * from emp;

Display all records.

SELECT Statement

- Used for queries on single or multiple tables
- Clauses of the SELECT statement:

SELECT

- List the columns (and expressions) to be returned from the query

FROM

- Indicate the table(s) or view(s) from which data will be obtained

WHERE

- Indicate the conditions under which a row will be included in the result

GROUP BY

- Indicate categorization of results

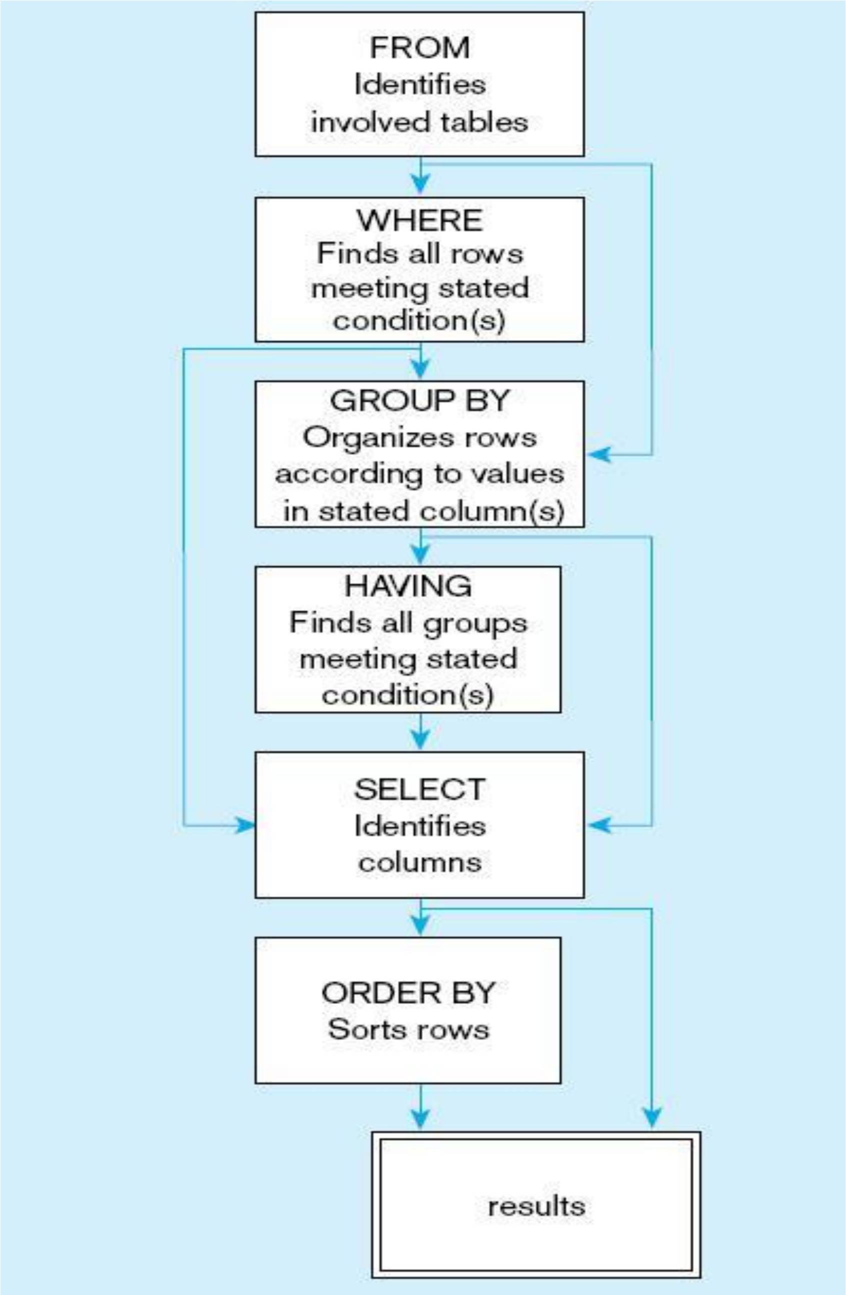
HAVING

- Indicate the conditions under which a category (group) will be included

ORDER BY

- Sorts the result according to specified criteria

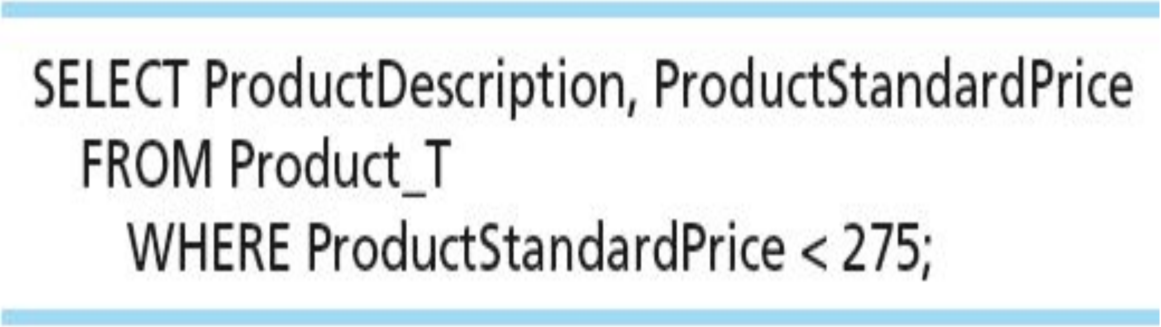
SQL statement processing



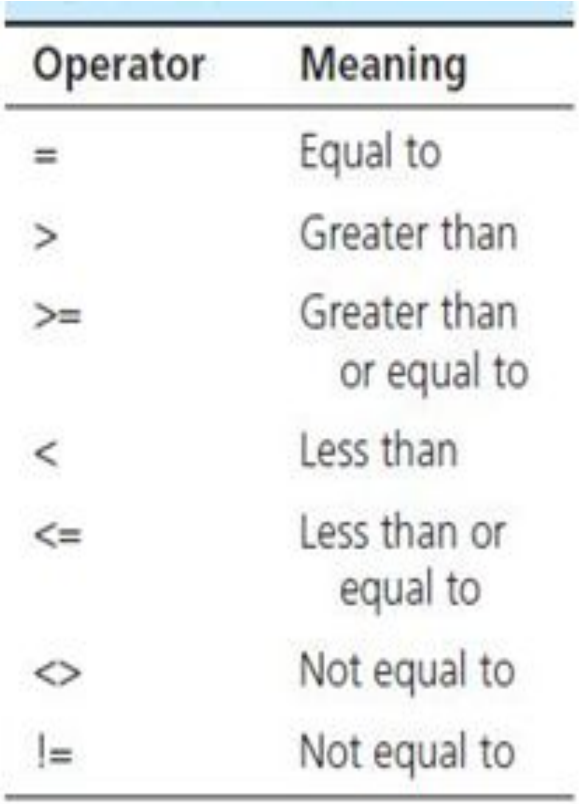
FROM
Identifies involved tables
WHERE
Finds all rows meeting stated condition(s)
GROUP BY
Organizes rows according to values in stated column(s)
HAVING
Finds all groups meeting stated condition(s)
SELECT
Identifies columns
ORDER BY
Sorts rows
results

SELECT Example

- Find products with standard price less than $275



```
SELECT ProductDescription, ProductStandardPrice
  FROM Product_T
    WHERE ProductStandardPrice < 275;
```



Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to
!=	Not equal to

Table: Comparison Operators in SQL

Example

```
SQL> select ename, salary, comm  from emp;
```

Display selected field only.

```
SQL> select * from emp where dno=10;
```

```
SQL> select salary+500 from emp where dno=10;
```

```
SQL> select * from emp where dno=10 and salary>5000;
```

SQL - Other Features

- DISTINCT
- Arithmetic operators: +, -, *, /
- Comparison operators: =, >, >=, <, <=, <>
- Concatenation operator: ||
- Substring comparisons: %, _
- BETWEEN
- AND, OR, NOT

- SQL> select dno from dept where dname=‘sales’;
- SQL> select * from emp where dno=10;
- SQL> select * from emp where dno=(select dno from dept where dname=‘sales’);
- SQL> select * from emp where dno in (select dno from dept where loc=‘salem’);
- SQL> select ename, salary from emp where salary = (select max(salary) from emp);
- SQL> select ename, salary from emp where salary > (select avg(salary) from emp);

in

- SQL> select ename, salary from emp where job=‘salesman’ or job=‘manager’;

- SQL> select ename, salary from emp where job in (‘manager’ , ‘salesman’);

Not in

- SQL> select ename, salary from emp where job notin (‘manager’ , ‘salesman’);

The **LIKE** operator

```
SELECT   *
FROM      Products
WHERE    PName LIKE ‘%gizmo%’
```

- s **LIKE** p: pattern matching on strings
- p may contain two special symbols:
  - % = any sequence of characters
  - _ = any single character

like

- SQL> select * from emp where ename like ‘Raj’;
- SQL> select * from emp where ename notlike ‘Raj’;

between

- SQL> select * from emp where salary <3000 or salary>10000;
- SQL> select * from emp where salary between 5000 and 10000;
- SQL> select * from emp where salary notbetween 5000 and 10000;

Eliminating Duplicates

```
SELECT   DISTINCT category
FROM     Product
```

Category
Gadgets
Photography
Household

Compare to:

```
SELECT   category
FROM     Product
```

Category
Gadgets
Gadgets
Photography
Household

Ordering the Results

```
SELECT   pname, price, manufacturer
FROM     Product
WHERE    category='gizmo' AND price > 50
ORDER BY  price
```

Ordering is ascending, unless you specify the DESC keyword.

PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

```
SELECT   DISTINCT category
FROM     Product
ORDER BY category
```

⇨ ?

```
SELECT   Category
FROM     Product
ORDER BY  PName
```

⇨ ?

```
SELECT   DISTINCT category
FROM     Product
ORDER BY PName
```

⇨ ?

Keys and Foreign Keys

Company

Key

CName	StockPrice	Country
GizmoWorks	25	USA
Canon	65	Japan
Hitachi	15	Japan

Product

PName	Price	Category	CName
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

Foreign key

WHERE Conditions

```
SELECT * FROM COURSE
WHERE COURSE_CODE LIKE ‘MIS%’;
```

USE % TO SUBSTITUTE FOR ANY STRING

```
SELECT * FROM COURSE
WHERE CREDIT HOURS BETWEEN 3 AND 5;
```

3 AND 5 ARE INCLUDED

```
SELECT * FROM CUSTOMER
WHERE BALANCE < CREDIT_LIMIT;
```

YOU CAN COMPARE TWO COLUMNS

More WHERE Conditions

```
SELECT * FROM CUSTOMER
WHERE STATE IN (‘OH’,’WV’,’KY’);
```

LIST OF SPECIFIC VALUES TO LOOK FOR

```
SELECT * FROM CUSTOMER
WHERE (CREDIT_LIMIT - BALANCE) <1000;
```

CAN MANIPULATE NUMBER VALUES MATHMATICALLY

AND/OR/NOT Conditions

```
SELECT * FROM CUSTOMER
WHERE BALANCE >=500
  AND BALANCE<=1000;
```

TWO COMPARISONS ON THE SAME COLUMN

```
SELECT * FROM CUSTOMER
WHERE STATE = ‘OH’
  OR CREDIT_LIMIT>1000;
```

TWO COMPARISONS ON THE DIFFERENT COLUMNS

```
SELECT * FROM CUSTOMER
WHERE NOT (STATE=‘OH’);
```

SAME AS STATE<>‘OH’

More on AND/OR/NOT

```
SELECT * FROM CUSTOMER
WHERE STATE = ‘OH’
  OR (CREDIT_LIMIT=1000
  AND BALANCE <500);
```

Use parentheses to make complex logic more understandable.

CUST	STATE	LIMIT	BAL
A	OH	1000	600
B	WV	1000	200
C	OH	500	300
D	OH	1000	200
E	KY	1300	800
F	KY	1000	700
G	MA	200	100
H	NB	1000	100

Who will be selected??

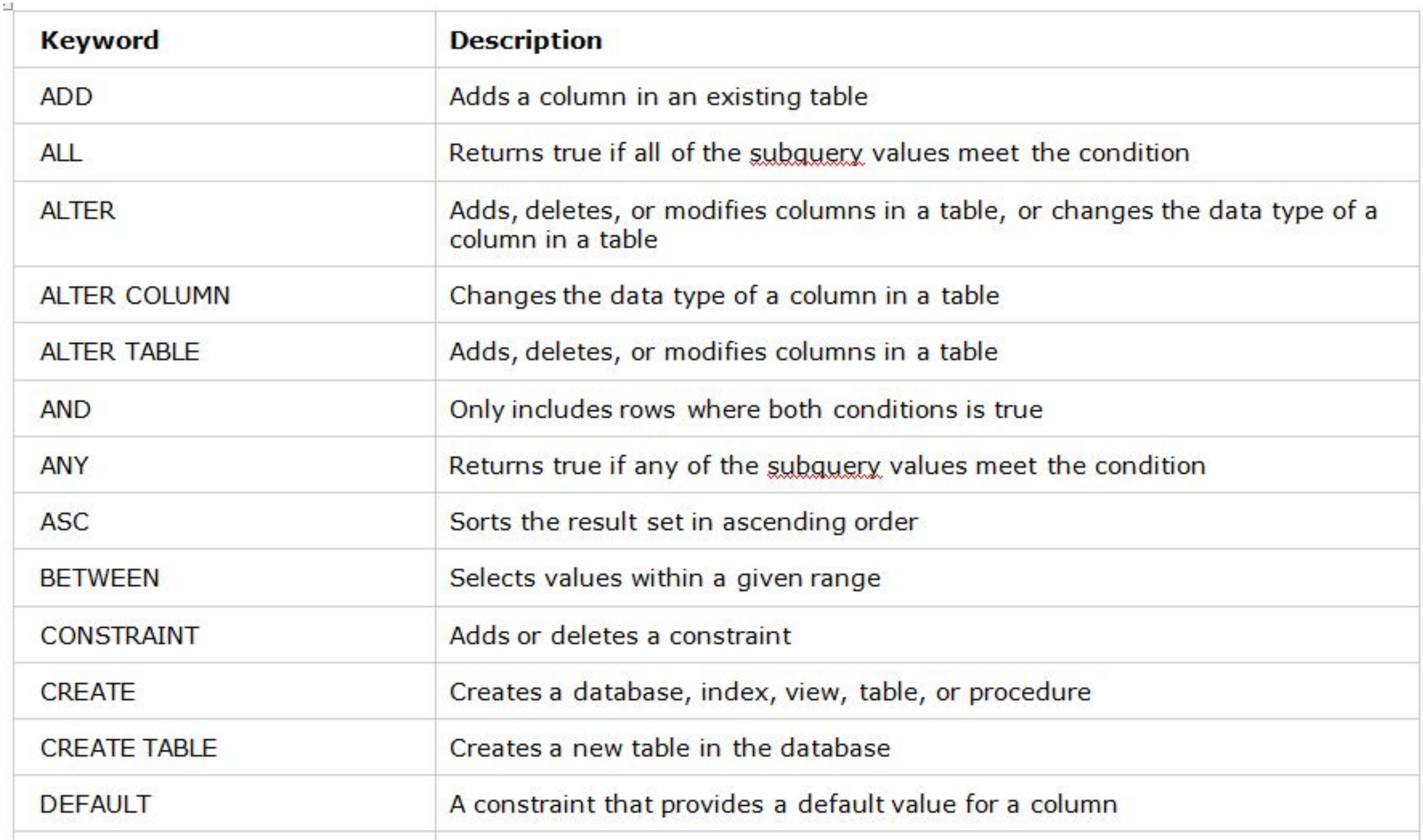
SQL for Retrieving Data from Two or More Tables

SQL provides two ways to retrieve data from related tables:

- Join - When two or more tables are joined by a common field.

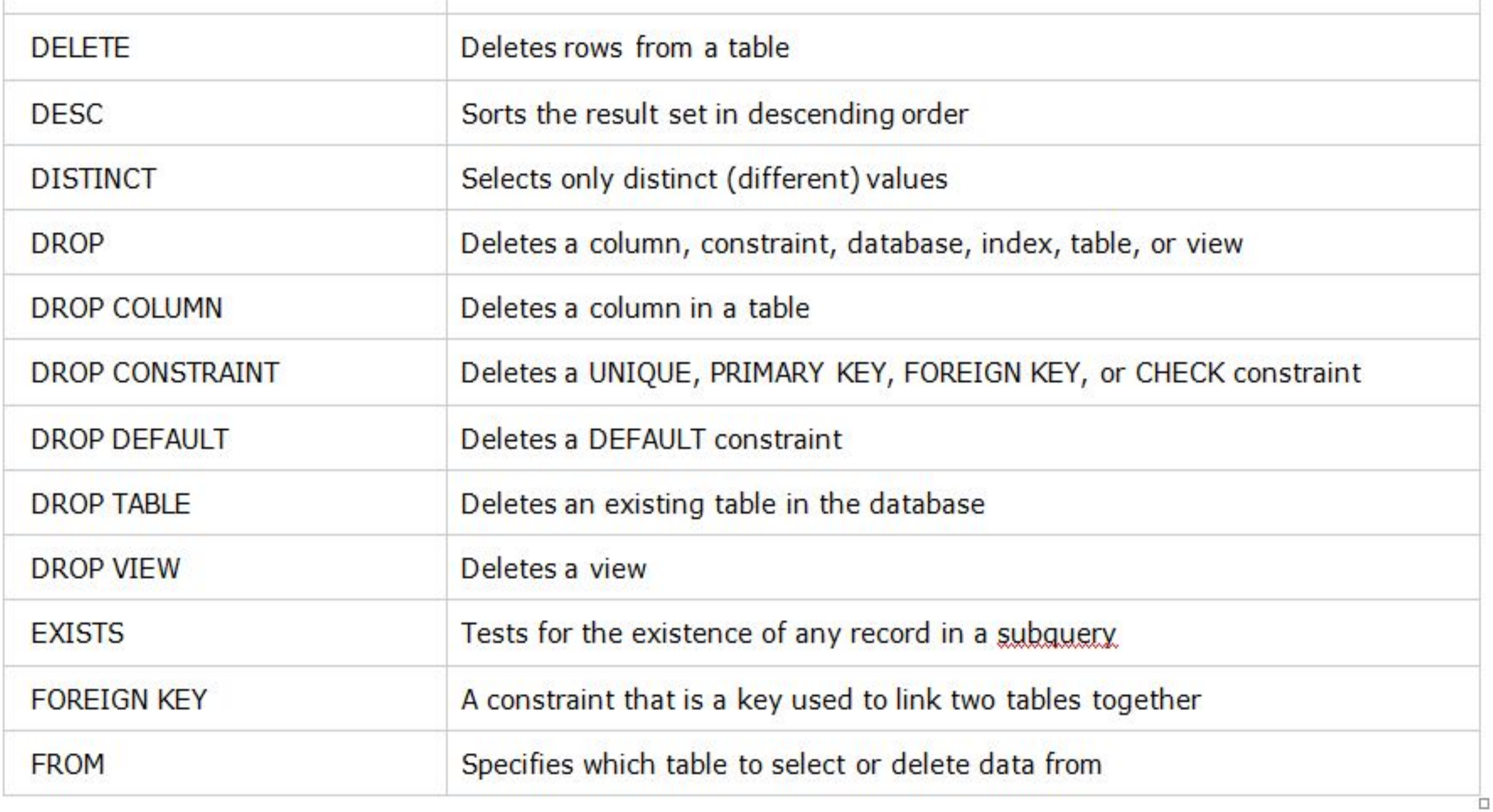
- Subqueries - When one Select command is nested within another command.

SQL Keywords



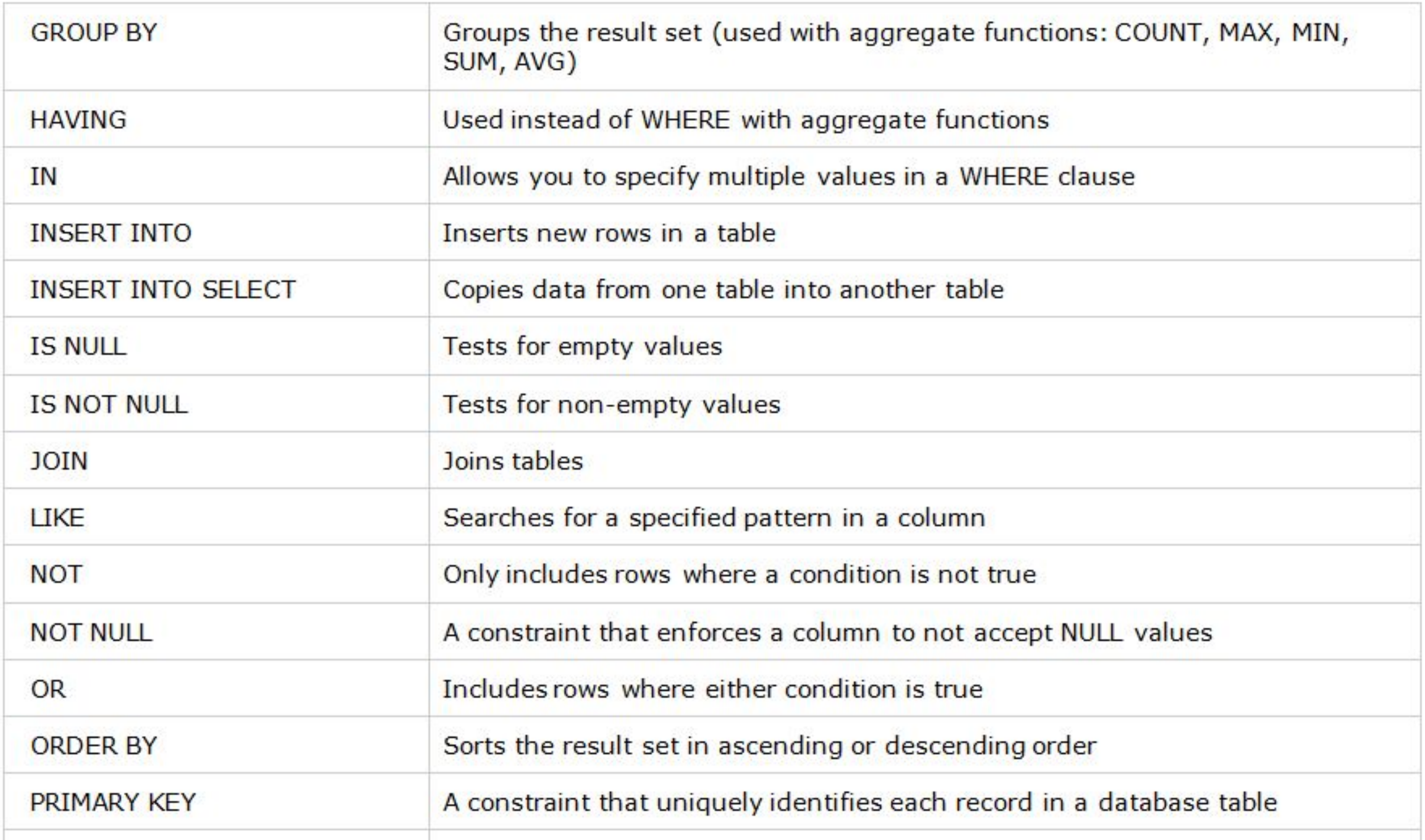
Keyword	Description
ADD	Adds a column in an existing table
ALL	Returns true if all of the subquery values meet the condition
ALTER	Adds, deletes, or modifies columns in a table, or changes the data type of a column in a table
ALTER COLUMN	Changes the data type of a column in a table
ALTER TABLE	Adds, deletes, or modifies columns in a table
AND	Only includes rows where both conditions is true
ANY	Returns true if any of the subquery values meet the condition
ASC	Sorts the result set in ascending order
BETWEEN	Selects values within a given range
CONSTRAINT	Adds or deletes a constraint
CREATE	Creates a database, index, view, table, or procedure
CREATE TABLE	Creates a new table in the database
DEFAULT	A constraint that provides a default value for a column

SQL Keywords



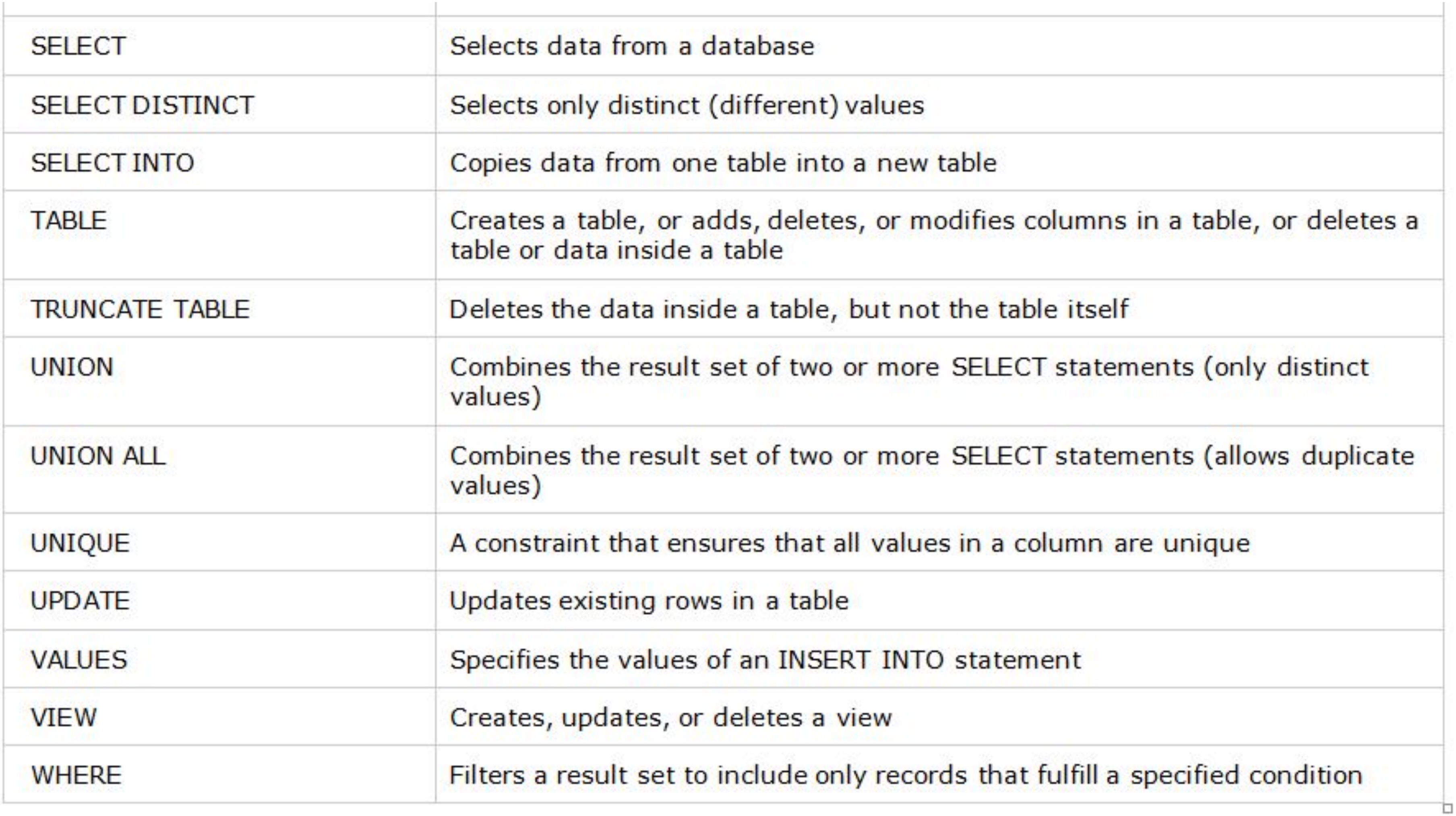
DELETE	Deletes rows from a table
DESC	Sorts the result set in descending order
DISTINCT	Selects only distinct (different) values
DROP	Deletes a column, constraint, database, index, table, or view
DROP COLUMN	Deletes a column in a table
DROP CONSTRAINT	Deletes a UNIQUE, PRIMARY KEY, FOREIGN KEY, or CHECK constraint
DROP DEFAULT	Deletes a DEFAULT constraint
DROP TABLE	Deletes an existing table in the database
DROP VIEW	Deletes a view
EXISTS	Tests for the existence of any record in a subquery
FOREIGN KEY	A constraint that is a key used to link two tables together
FROM	Specifies which table to select or delete data from

SQL Keywords



GROUP BY	Groups the result set (used with aggregate functions: COUNT, MAX, MIN, SUM, AVG)
HAVING	Used instead of WHERE with aggregate functions
IN	Allows you to specify multiple values in a WHERE clause
INSERT INTO	Inserts new rows in a table
INSERT INTO SELECT	Copies data from one table into another table
IS NULL	Tests for empty values
IS NOT NULL	Tests for non-empty values
JOIN	Joins tables
LIKE	Searches for a specified pattern in a column
NOT	Only includes rows where a condition is not true
NOT NULL	A constraint that enforces a column to not accept NULL values
OR	Includes rows where either condition is true
ORDER BY	Sorts the result set in ascending or descending order
PRIMARY KEY	A constraint that uniquely identifies each record in a database table

SQL Keywords



SELECT	Selects data from a database
SELECT DISTINCT	Selects only distinct (different) values
SELECT INTO	Copies data from one table into a new table
TABLE	Creates a table, or adds, deletes, or modifies columns in a table, or deletes a table or data inside a table
TRUNCATE TABLE	Deletes the data inside a table, but not the table itself
UNION	Combines the result set of two or more SELECT statements (only distinct values)
UNION ALL	Combines the result set of two or more SELECT statements (allows duplicate values)
UNIQUE	A constraint that ensures that all values in a column are unique
UPDATE	Updates existing rows in a table
VALUES	Specifies the values of an INSERT INTO statement
VIEW	Creates, updates, or deletes a view
WHERE	Filters a result set to include only records that fulfill a specified condition

Questions

1. Give syntax and apply the DDL and DML commands for defining and constructing two tables of your choice with appropriate data.

2. Illustrate different Integrity constraints in relational model with appropriate examples

3. Explain Relational Query languages with proper Examples.

4. Give syntax and apply the SQL commands for defining two example tables of your choice. Then insert data, update data in the tables

Questions

5. Explain Primary and Foreign Key constraints with examples.

6. Explain Relation Algebra operators with examples

7.Consider the following schema of a bank database.

Branches (B_name: string, B_city:string, Assets_Value:integer)
Accounts(Acc_No:string, B_name:string, Balance:real)
Loans(L_No:string, B_name:sring, Amount:real).

Create tables for the above schema with primary key, foreign key and not null constraints wherever necessary.

Questions

8. Consider the following schema and answer the following queries in SQL.

   Suppliers(sid:integer, sname:string, address:string)

   Parts(pid:integer, pname:string, colour:string)

   Catalog(sid:integer, pid:integer, cost:real)

i)Find the ids and names of suppliers whose name begins with character “A”.

ii)Find the ids of suppliers who supply some Red colour part.

iii)Find the ids of suppliers who supply either a Red colour part or a Green colour part.

iv)Find the names of all parts supplied by supplier named John.

Questions

9. Consider the following schema and answer the following queries in SQL.

Students(sid: string, sname:string, Date_of_Birth:date, GPA:real)

Courses(cid:string, cname:string, credits:integer, offered_by:string)

Enrolled(sid:string, cid:string)

i) For each course offered by CSE department, find the total number of enrollments.

ii) Find the sum of credits of all courses taken by student “ S01”.

iii) Find the courses that have at least 10 enrollments.

Questions

10. Based on the below schemas construct the corresponding SQL queries

Sailors (sid:string, sname:string, rating:integer, age:real)

Boats (bid:integer, bname:string, color:string)

Reserves (sid:integer, bid:integer, day:date)

- Find the colors of boats reserved by Lubber.
- Find the names of sailors who have reserved at least one boat.
- Find the names of sailors who have reserved both a red and a green boat.
- Find the names of sailors who have reserved all boats.
- Display the names of sailors whose name starts with “S”.
- Display all the sailor’s names alphabetical order.
- Display the sid and number of boats reserved by each sailor

- What is the full form of SQL?

1. Structured Query List
2. Structure Query Language
3. Sample Query Language
4. None of these.

- Which of the following is not a valid SQL type?

1. FLOAT
2. NUMERIC
3. DECIMAL
4. CHARACTER

- Which of the following is not a DDL command?

1. TRUNCATE
2. ALTER
3. CREATE
4. UPDATE

- Which of the following are TCL commands?

1. COMMIT and ROLLBACK
2. UPDATE and TRUNCATE
3. SELECT and INSERT
4. GRANT and REVOKE

- SQL Views are also known as

1. Simple tables
2. Virtual tables
3. Complex tables
4. Actual Tables

- If we have not specified ASC or DESC after a SQL ORDER BY clause, the following is used by default

1. DESC
2. ASC
3. There is no default value
4. None of the mentioned

- Which of the following is not Constraint in SQL?

1. Primary Key
2. Not Null
3. Check
4. Union

- Which of the following is not a valid aggregate function?

1. COUNT
2. COMPUTE
3. SUM
4. MAX

- Which data manipulation command is used to combines the records from one or more tables?

1. SELECT
2. PROJECT
3. JOIN
4. PRODUCT

- Which operator is used to compare a value to a specified list of values?

1. ANY
2. BETWEEN
3. ALL
4. IN

- What operator tests column for absence of data

1. NOT Operator
2. Exists Operator
3. IS NULL Operator
4. None of the above

- In which of the following cases a DML statement is not executed?

1. When existing rows are modified.
2. When a table is deleted.
3. When some rows are deleted.
4. All of the above

- A command that lets you change one or more field in a table is:

1. INSERT
2. MODIFY
3. LOOK-UP
4. All of the abov

- What is returned by

INSTR(‘SRKR Engineering College’, ‘C’)?

1. 6
2. 18
3. C
4. Engineering

Which of the following is also called an INNER JOIN

1. EQUI JOIN
2. SELF JOIN
3. NON-EQUI JOIN
4. None of the above

Which of the following is true about the HAVING clause?

1. Similar to the WHERE clause but is used for columns rather than groups.

2. Similar to WHERE clause but is used for rows rather than columns.

3. Similar to WHERE clause but is used for groups rather than rows.

4. Acts exactly like a WHERE clause.

- __________ command makes the updates performed by the transaction permanent in the database?

1. ROLLBACK
2. COMMIT
3. TRUNCATE
4. DELETE

How can you change "Thomas" into "Michel" in the "LastName" column in the Users table?

1. UPDATE User SET LastName = 'Thomas' INTO LastName = 'Michel’

2. MODIFY Users SET LastName = 'Michel' WHERE LastName = 'Thomas’

3. MODIFY Users SET LastName = 'Thomas' INTO LastName = 'Michel’

4. UPDATE Users SET LastName = 'Michel' WHERE LastName = 'Thomas'

Which command is used to change the definition of a table in SQL?

1. CREATE
2. UPDATE
3. ALTER
4. SELECT

Which type of JOIN is used to returns rows that do not have matching values?

1. Natural JOIN
2. Outer JOIN
3. EQUI JOIN
4. All of the above

Which of the following is the basic approaches for joining tables?

1. Union JOIN
2. Natural JOIN
3. Subqueries
4. All of the above

Why we need to create an index if the primary key is already present in a table?

1. Index improves the speed of data retrieval operations on a table.
2. Indexes are special lookup tables that will be used by the database search engine.
3. Indexes are synonyms of a column in a table.
4. All of the above

Group of operations that form a single logical unit of work is known as

1. View
2. Network
3. Unit
4. Transaction

- Which of the following is the correct order of a SQL statement?

1. SELECT, GROUP BY, WHERE, HAVING
2. SELECT, WHERE, GROUP BY, HAVING
3. SELECT, HAVING, WHERE, GROUP BY
4. SELECT, WHERE, HAVING, GROUP BY