UNITED
Group of Institutions
Greater Noida • Allahabad

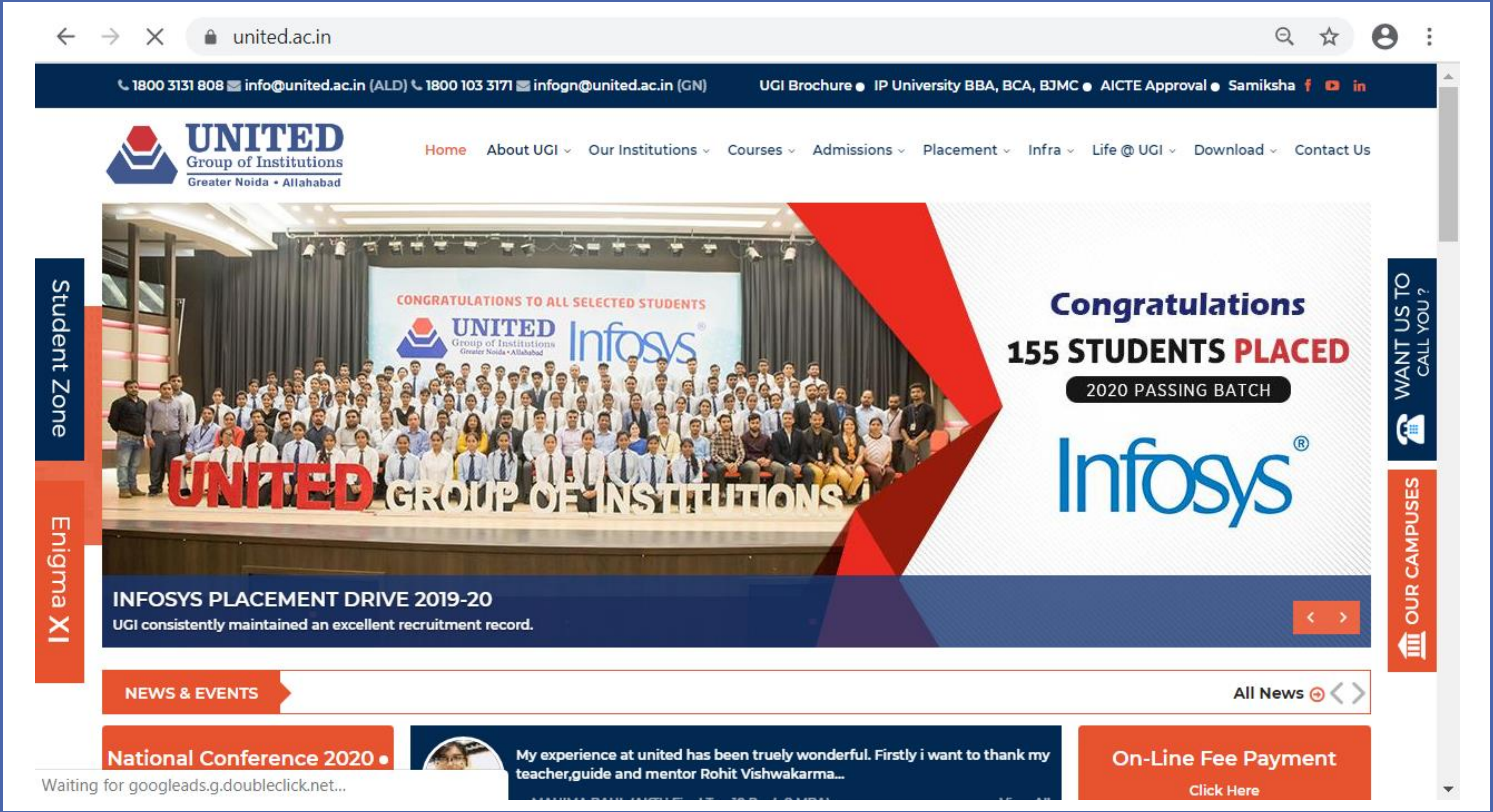
SQL

Presented by : **Prashant Soni** ‘Technical Trainer, UCER‘

Objectives

After completing this session, you should be able to do the following:

- Perform the various types of SQL operations:
  - DDL (Data Definition Language) Queries
  - DQL (Data Query Language) Queries
  - DML (Data Manipulation Language) Queries
  - DCL (Data Control Language) Queries
- Able to answer the interview’s important questions based on SQL.


united.ac.in

1800 3131 808 info@united.ac.in (ALD) 1800 103 3171 infogn@united.ac.in (GN) UGI Brochure ● IP University BBA, BCA, BJMC ● AICTE Approval ● Samiksha

UNITED Group of Institutions Greater Noida • Allahabad

Home About UGI Our Institutions Courses Admissions Placement Infra Life @ UGI Download Contact Us

Student Zone

Enigma XI

WANT US TO CALL YOU ?

OUR CAMPUSES

CONGRATULATIONS TO ALL SELECTED STUDENTS
UNITED Group of Institutions
Infosys
UNITED GROUP OF INSTITUTIONS
Congratulations
155 STUDENTS PLACED
2020 PASSING BATCH
Infosys

INFOSYS PLACEMENT DRIVE 2019-20

UGI consistently maintained an excellent recruitment record.

NEWS & EVENTS

All News

National Conference 2020 ●

My experience at united has been truely wonderful. Firstly i want to thank my teacher,guide and mentor Rohit Vishwakarma...

On-Line Fee Payment

Click Here

Waiting for googleads.g.doubleclick.net...

IMPORTANT

Q1. What is the difference between Front and Back End? (Asked by Infosys, TCS, Wipro, etc. in 2018,19)

Q2. What technology did you used as front End and Backend in Your project? (Asked by Infosys 2017,18,19)

Important Part of Any Website

Front End

- It is the visible part of the web site
- Human or digital users interact directly with various aspects of the **front end** of a website.

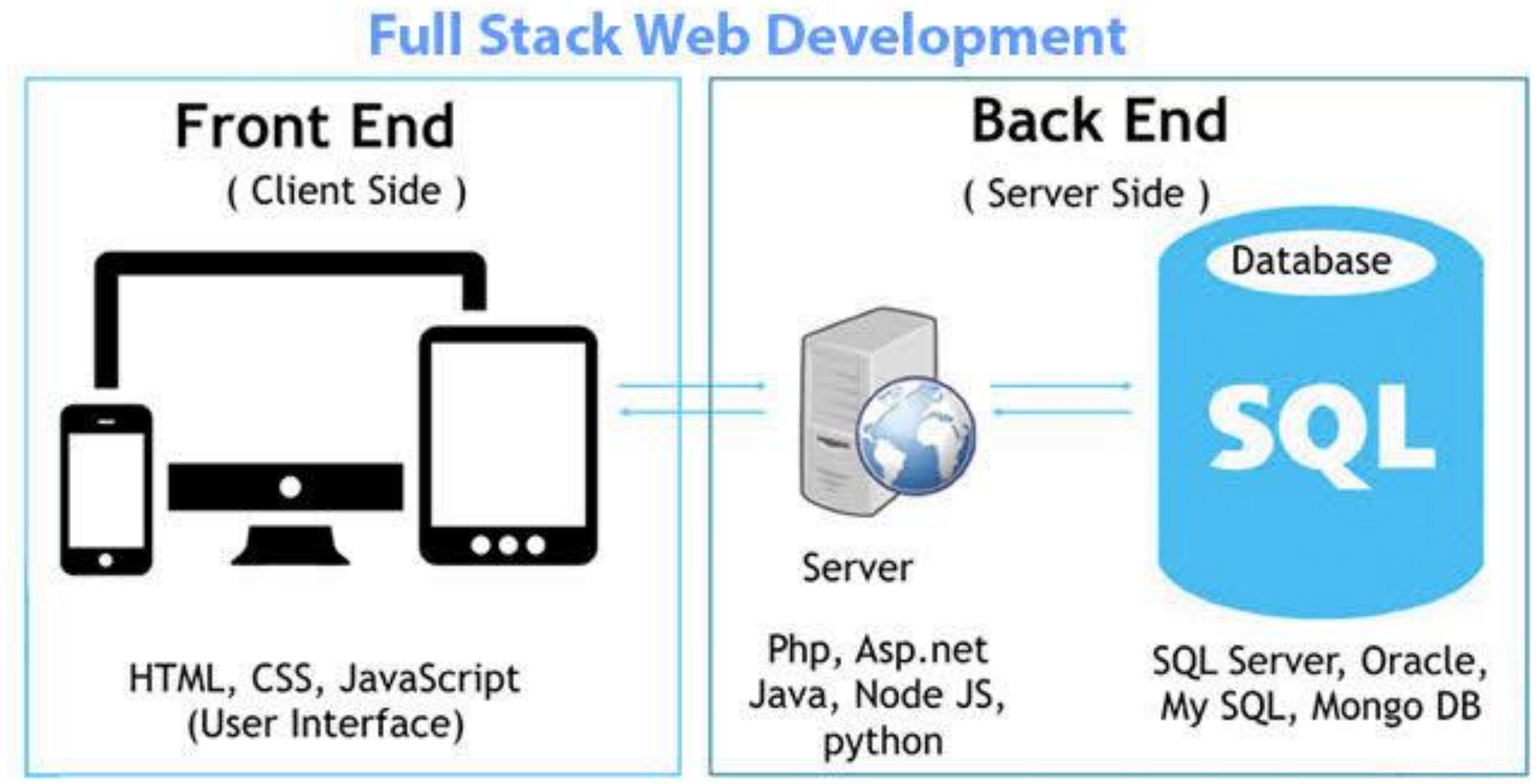
Back End

- It is the knowledge base behind the front end
- It comprises three components: server, application, and database.

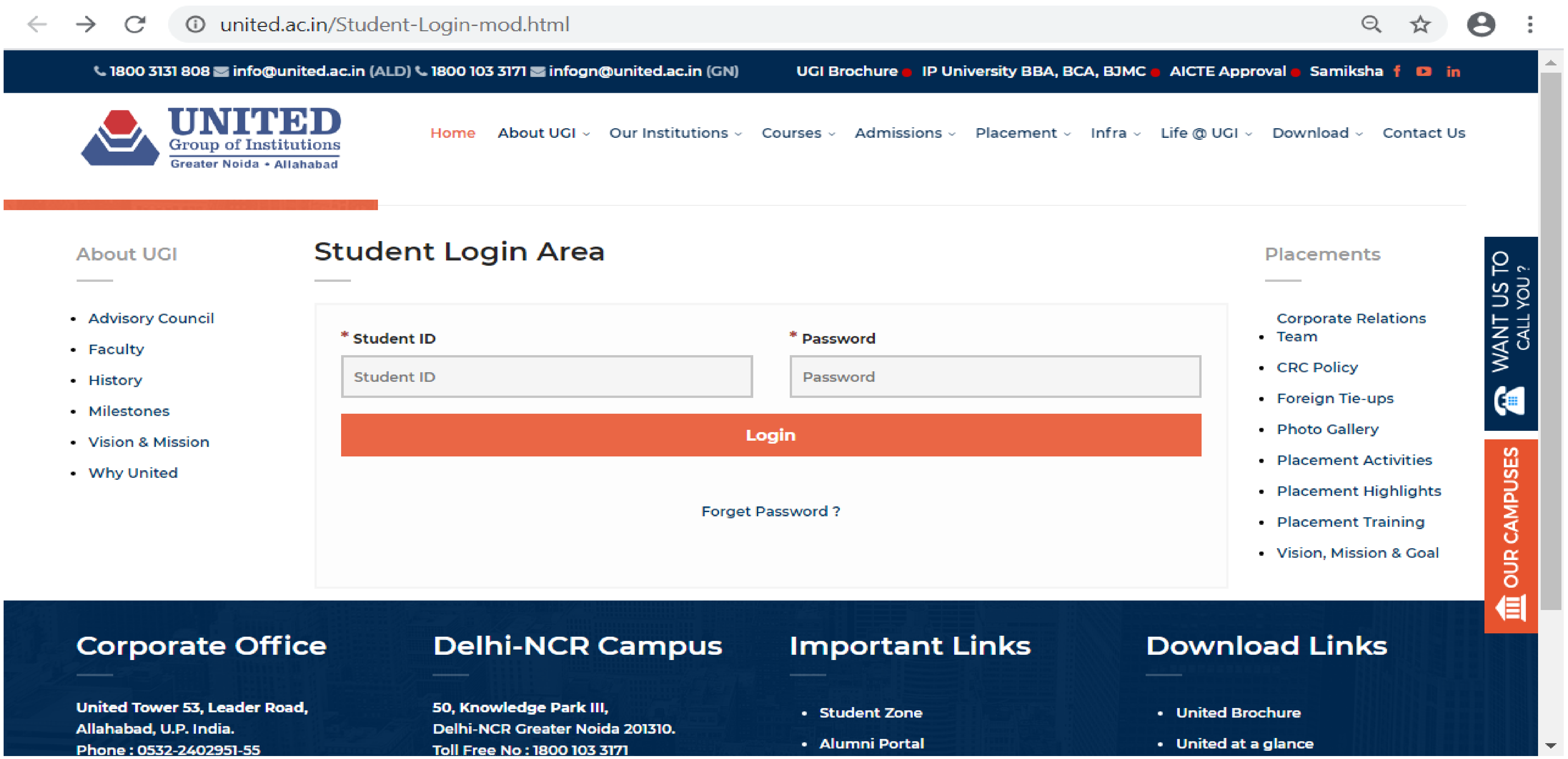
IMPORTANT

Q1.What is the difference between SQL and MYSQL? (Asked by INFOSYS, TCS in 2017,18,19)

Q2.What is the full form of PHP? (Asked by INFOSYS in 2019)



WHAT IS THE ROLE OF "SQL" IN ANY PROJECT



united.ac.in/Student-Login-mod.html

1800 3131 808 info@united.ac.in (ALD) 1800 103 3171 infogn@united.ac.in (GN) UGI Brochure ● IP University BBA, BCA, BJMC ● AICTE Approval ● Samiksha

UNITED Group of Institutions Greater Noida • Allahabad

Home | About UGI | Our Institutions | Courses | Admissions | Placement | Infra | Life @ UGI | Download | Contact Us

About UGI

- Advisory Council
- Faculty
- History
- Milestones
- Vision & Mission
- Why United

Student Login Area

* Student ID

Student ID

* Password

Password

Login

Forget Password ?

Placements

- Corporate Relations Team
- CRC Policy
- Foreign Tie-ups
- Photo Gallery
- Placement Activities
- Placement Highlights
- Placement Training
- Vision, Mission & Goal

WANT US TO CALL YOU ?

OUR CAMPUSES

Corporate Office

United Tower 53, Leader Road,
Allahabad, U.P. India.
Phone : 0532-2402951-55

Delhi-NCR Campus

50, Knowledge Park III,
Delhi-NCR Greater Noida 201310.
Toll Free No : 1800 103 3171

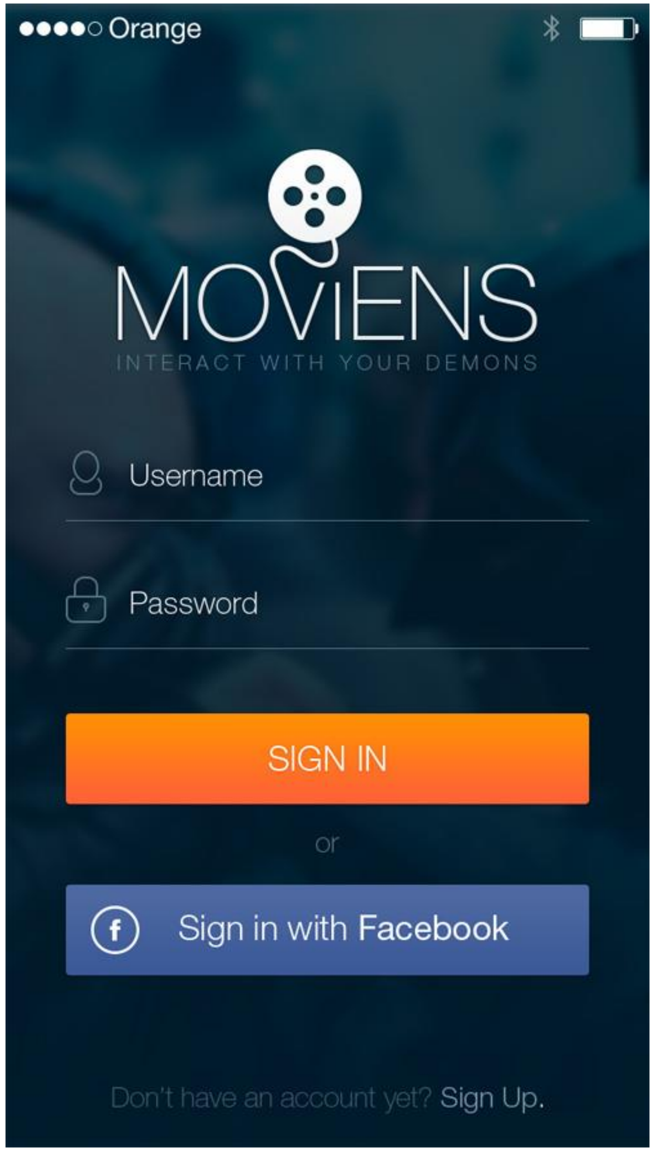
Important Links

- Student Zone
- Alumni Portal

Download Links

- United Brochure
- United at a glance

WHAT IS THE ROLE OF "**SQL**" IN ANY PROJECT



IMPORTANT

Q1.Full form of SQL? (Asked by INFOSYS, TCS in 2017,18,19)

Q2.Why did we say that SQL is a universal language? (Asked by INFOSYS in 2019)

Introduction

- **SQL** stands for Structured Query Language
- **SQL** was initially **developed** at IBM by Donald D. Chamberlin and Raymond F in 1970
- **SQL** is used to communicate with a database
- **SQL** is a **non-procedural language**
- **IN SQL** we need to only describe what we '**want to be done'.**
- Also, they are using different dialects, such as –
  - MS SQL Server using T-SQL,
  - Oracle using PL/SQL,
  - MS Access version of SQL is called JET SQL (native format) etc.

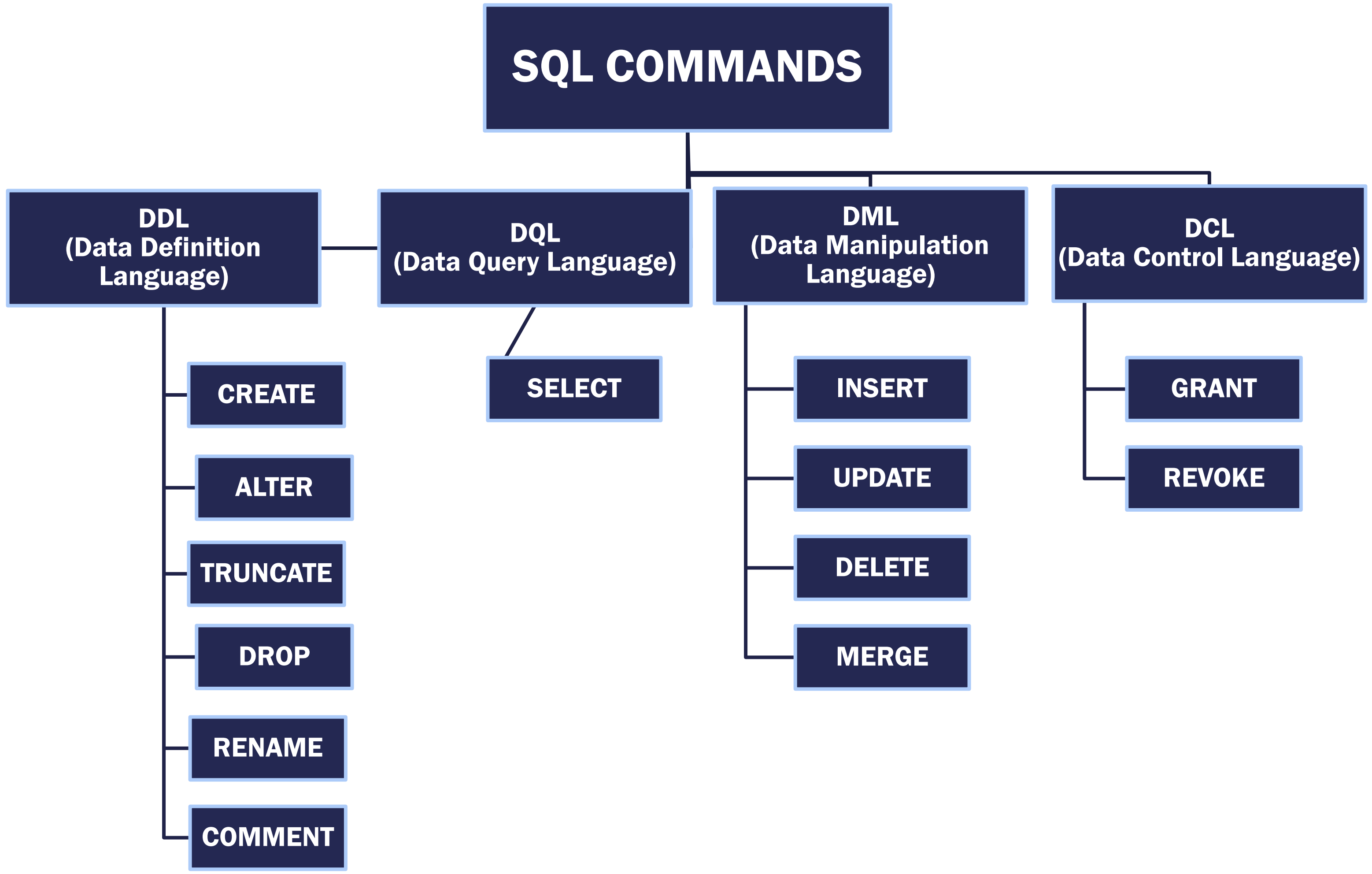
IMPORTANT

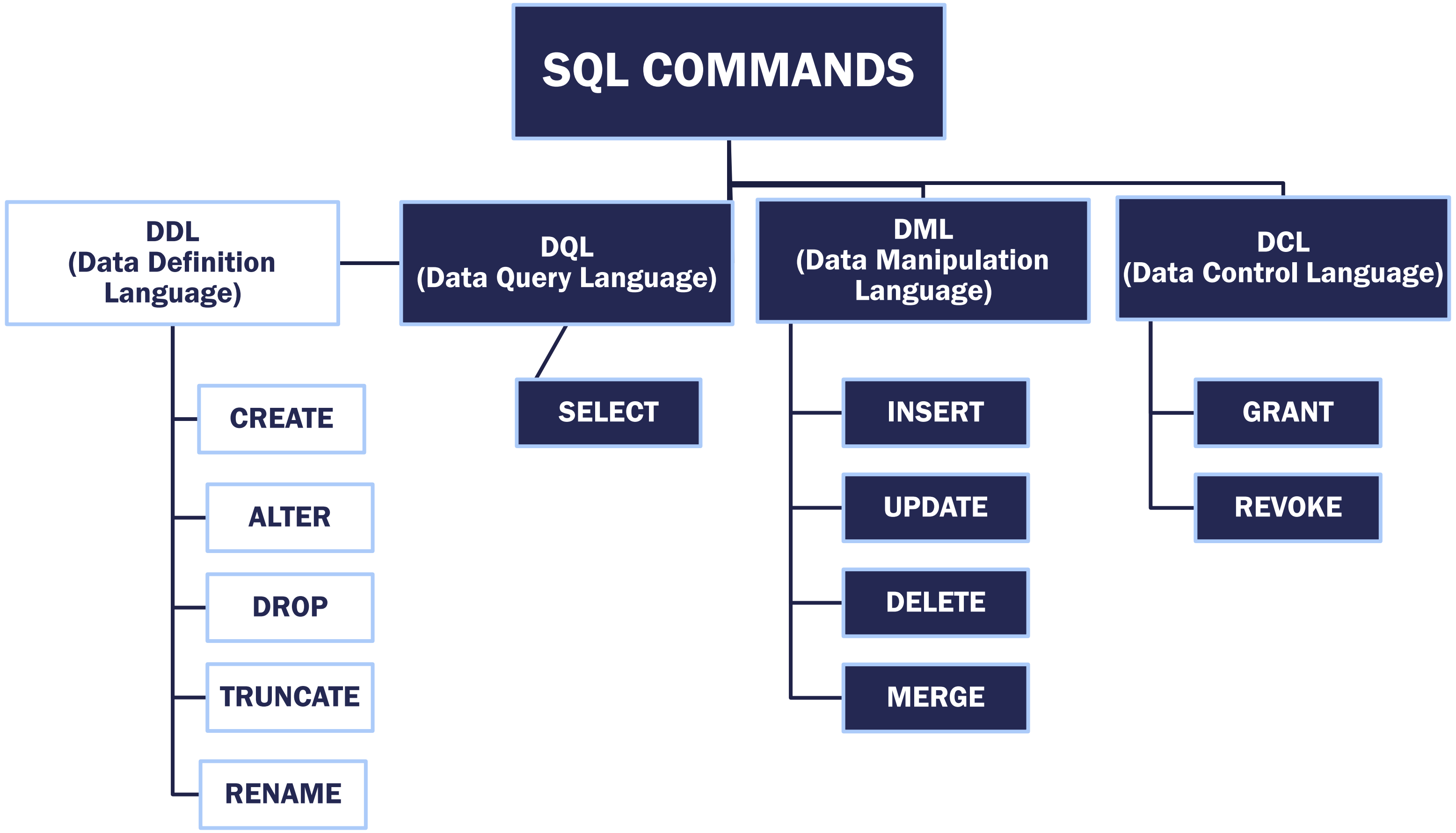
Q1.What is the difference between DDL and DML ? (Asked by INFOSYS, TCS in 2017,18,19)

Q2.What is schema? (Asked by MIND in 2019)

SQL Environment

- 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 Query Language (DQL)
  - A set of schemas that constitute the description of a database
- 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


SQL COMMANDS
DDL (Data Definition Language)
CREATE
ALTER
TRUNCATE
DROP
RENAME
COMMENT
DQL (Data Query Language)
SELECT
DML (Data Manipulation Language)
INSERT
UPDATE
DELETE
MERGE
DCL (Data Control Language)
GRANT
REVOKE


SQL COMMANDS
DDL
(Data Definition Language)
DQL
(Data Query Language)
DML
(Data Manipulation Language)
DCL
(Data Control Language)
CREATE
ALTER
DROP
TRUNCATE
RENAME
SELECT
INSERT
UPDATE
DELETE
MERGE
GRANT
REVOKE

DDL COMMANDS…

- The Create Command to create table

SYNTAX

```
Create table TableName(clo1 datatype(size), clo2 datatype(size)[CONSTRAINT],……. clon datatype(size));
```

EXAMPLE

```
Create table Demo( FirstName varchar2(20),LastName varchar2(20),salary number(5));
```

DDL COMMANDS…

- The ALTER Command is used to:-
  - Adding New Columns
  - Dropping a Column from the Table
  - Modifying Existing Table
  - Adding Constraints in Table

DDL COMMANDS…

- **The ALTER Command to add new column in table**

SYNTAX

```
ALTER table TableName add ColumnName datatype(size);
```

EXAMPLE

```
ALTER table Demo add dept number(2);
```

DDL COMMANDS…

- The ALTER Command to Remove a column from table

SYNTAX

```
Alter Table TableName  drop Column ColumnName;
```

EXAMPLE

```
Alter Table Demo drop column dept;
```

DDL COMMANDS…

- **The ALTER Command to modify existing column in table**

SYNTAX

```
Alter table TableName modify ColumnName newdatatype(newsize);
```

EXAMPLE

```
alter table Demo modify dept number(5);
```

DDL COMMANDS…

- The ALTER Command to add constraint for particular column

SYNTAX

```
Alter table TableName add CONSTRAINT constraintname constrainttype (ColumnName);
```

EXAMPLE

```
Alter table Demo add CONSTRAINT consprim  PRIMARY KEY (emp_id);
```

DDL COMMANDS…

- Drop command use to drop the table structure from database.

SYNTAX

```
Drop table TableName;
```

EXAMPLE

```
Drop table Demo;
```

DDL COMMANDS…

- **Rename command is use to rename the table in the database..**

SYNTAX

```
Alter Table table_name Rename to new_Table_Name
```

EXAMPLE

```
Alter table Demo Rename to Demo_emp;
```

DDL COMMANDS…

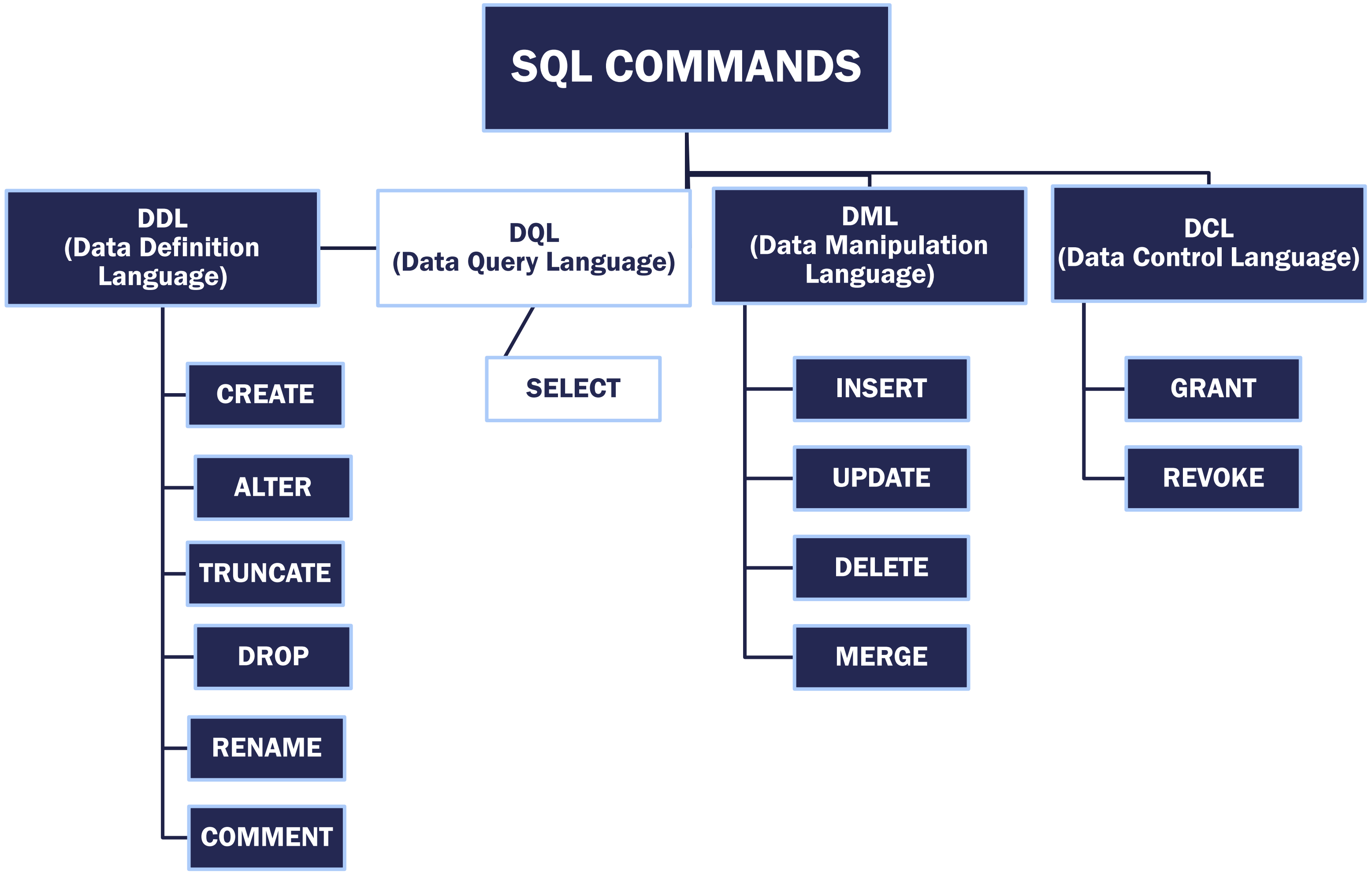
- Rename command is use to rename the column in the Table.

SYNTAX

```
Alter Table table_name Rename column old_column_name to new_column_Name
```

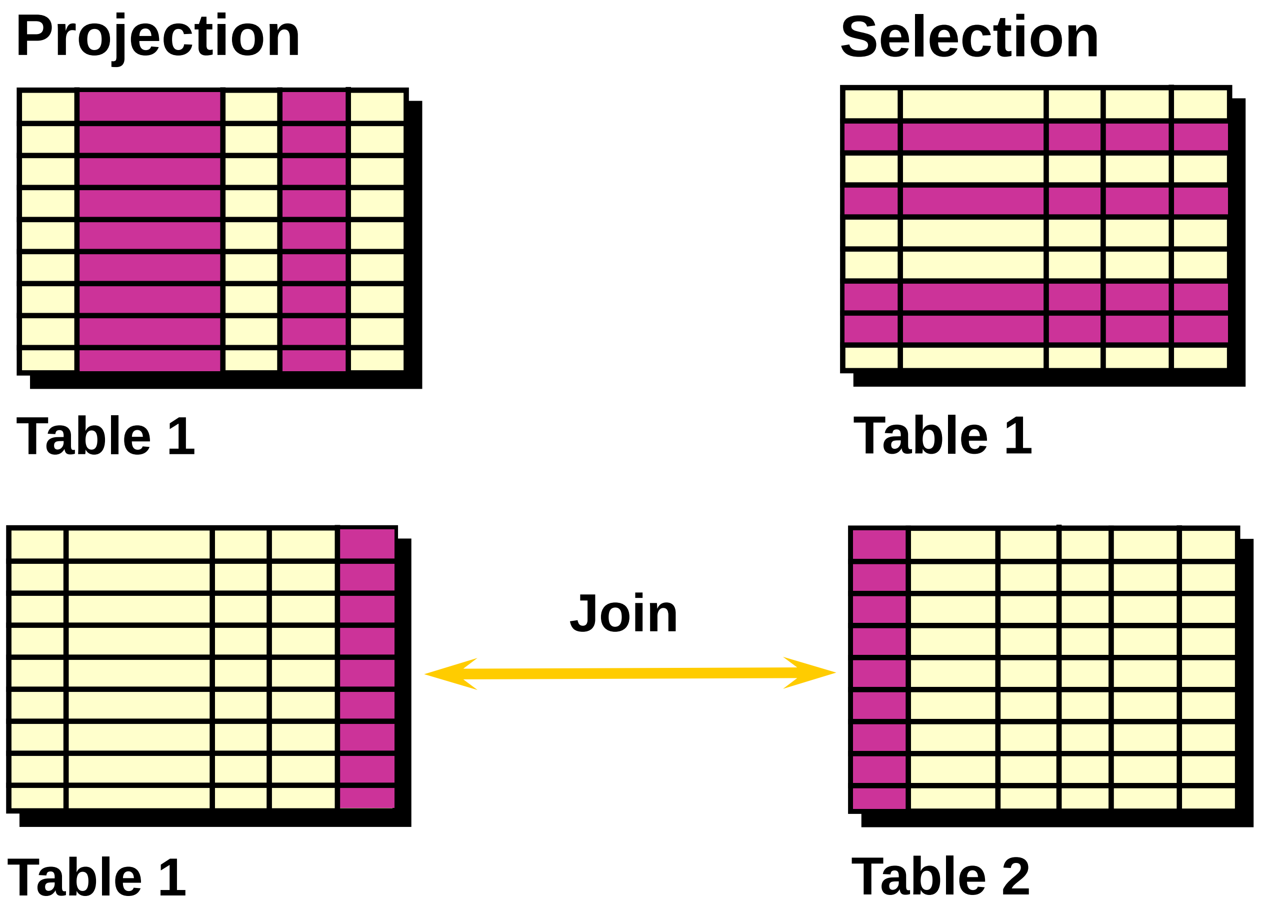
EXAMPLE

```
Alter table Demo Rename column dept  to dept_no;
```


SQL COMMANDS
DDL (Data Definition Language)
CREATE
ALTER
TRUNCATE
DROP
RENAME
COMMENT
DQL (Data Query Language)
SELECT
DML (Data Manipulation Language)
INSERT
UPDATE
DELETE
MERGE
DCL (Data Control Language)
GRANT
REVOKE

Writing Basic SQL SELECT Statements

Capabilities of SQL `SELECT` Statements



Basic SELECT Statement

```
SELECT   *|{[DISTINCT] column|expression [alias],...}
FROM     table;
```

- SELECT identifies *what* columns
- FROM identifies *which* table

Selecting All Columns

```
SELECT *
FROM   departments;
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting		1700

8 rows selected.

Selecting Specific Columns

```
SELECT department_id, location_id
FROM   departments;
```

DEPARTMENT_ID	LOCATION_ID
10	1700
20	1800
50	1500
60	1400
80	2500
90	1700
110	1700
190	1700

8 rows selected.

Writing SQL Statements

- SQL statements are not case sensitive.
- SQL statements can be on one or more lines.
- Keywords cannot be abbreviated or split across lines.
- Clauses are usually placed on separate lines.
- Indents are used to enhance readability.

Column Heading Defaults

- *i*SQL*Plus:
  - Default heading justification: Center
  - Default heading display: Uppercase

Arithmetic Expressions

Create expressions with number and date data by using arithmetic operators.

Operator	Description
+	Add
-	Subtract
*	Multiply
/	Divide

Using Arithmetic Operators

```
SELECT last_name, salary, salary + 300
FROM   employees;
```

LAST_NAME	SALARY	SALARY+300
King	24000	24300
Kochhar	17000	17300
De Haan	17000	17300
Hunold	9000	9300
Ernst	6000	6300

…

Hartstein	13000	13300
Fay	6000	6300
Higgins	12000	12300
Gietz	8300	8600

20 rows selected.

Operator Precedence

- Multiplication and division take priority over addition and subtraction.
- Operators of the same priority are evaluated from left to right.
- Parentheses are used to force prioritized evaluation and to clarify statements.

Operator Precedence

```
SELECT last_name, salary, 12*salary+100
FROM   employees;
```

LAST_NAME	SALARY	12*SALARY+100
King	24000	288100
Kochhar	17000	204100
De Haan	17000	204100
Hunold	9000	108100
Ernst	6000	72100

...

Hartstein	13000	156100
Fay	6000	72100
Higgins	12000	144100
Gietz	8300	99700

20 rows selected.

Using Parentheses

```
SELECT last_name, salary, 12*(salary+100)
FROM   employees;
```

LAST_NAME	SALARY	12*(SALARY+100)
King	24000	289200
Kochhar	17000	205200
De Haan	17000	205200
Hunold	9000	109200
Ernst	6000	73200

...

Hartstein	13000	157200
Fay	6000	73200
Higgins	12000	145200
Gietz	8300	100800

20 rows selected.

Defining a Null Value

- A null is a value that is unavailable, unassigned, unknown, or inapplicable.
- A null is not the same as zero or a blank space.

```
SELECT last_name, job_id, salary, commission_pct
FROM   employees;
```

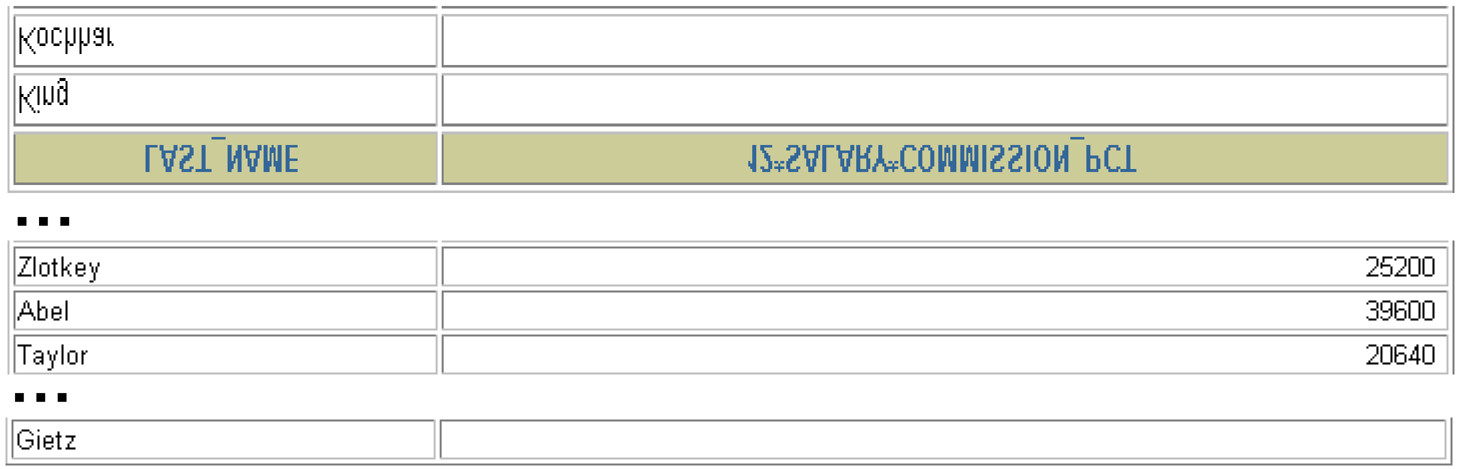
LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
King	AD_PRES	24000	
Kochhar	AD_VP	17000	
...			
Zlotkey	SA_MAN	10500	.2
Abel	SA_REP	11000	.3
Taylor	SA_REP	8600	.2
...			
Gietz	AC_ACCOUNT	8300	

20 rows selected.

Null Values in Arithmetic Expressions

Arithmetic expressions containing a null value evaluate to null.

```
SELECT last_name, 12*salary*commission_pct
FROM   employees;
```



LAST_NAME	12*SALARY*COMMISSION_PCT
King	
Kochhar	
...	
Zlotkey	25200
Abel	39600
Taylor	20640
...	
Gietz	

20 rows selected.

Defining a Column Alias

A column alias:

- Renames a column heading
- Is useful with calculations
- Immediately follows the column name - there can also be the optional `AS` keyword between the column name and alias
- Requires double quotation marks if it contains spaces or special characters or is case sensitive

Using Column Aliases

```
SELECT last_name AS name, commission_pct comm
FROM   employees;
```

NAME	COMM
King	
Kochhar	
De Haan	

...

20 rows selected.

```
SELECT last_name "Name", salary*12 "Annual Salary"
FROM   employees;
```

Name	Annual Salary
King	288000
Kochhar	204000
De Haan	204000

...

20 rows selected.

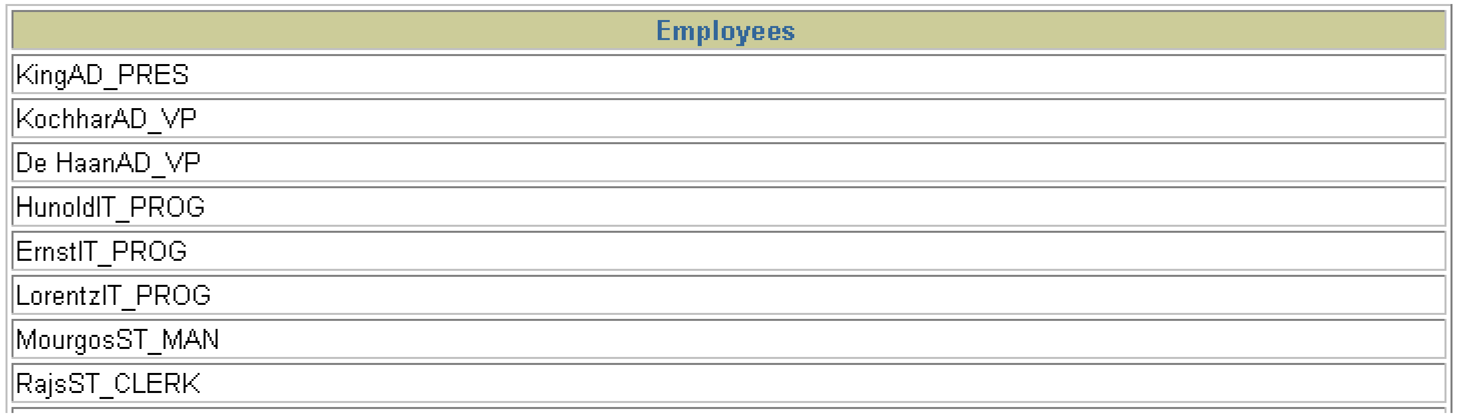
Concatenation Operator

A concatenation operator:

- Concatenates columns or character strings to other columns
- Is represented by two vertical bars (||)
- Creates a resultant column that is a character expression

Using the Concatenation Operator

```
SELECT    last_name||job_id AS "Employees"
FROM      employees;
```



Employees
KingAD_PRES
KochharAD_VP
De HaanAD_VP
HunoldIT_PROG
ErnstIT_PROG
LorentzIT_PROG
MourgosST_MAN
RajsST_CLERK

...

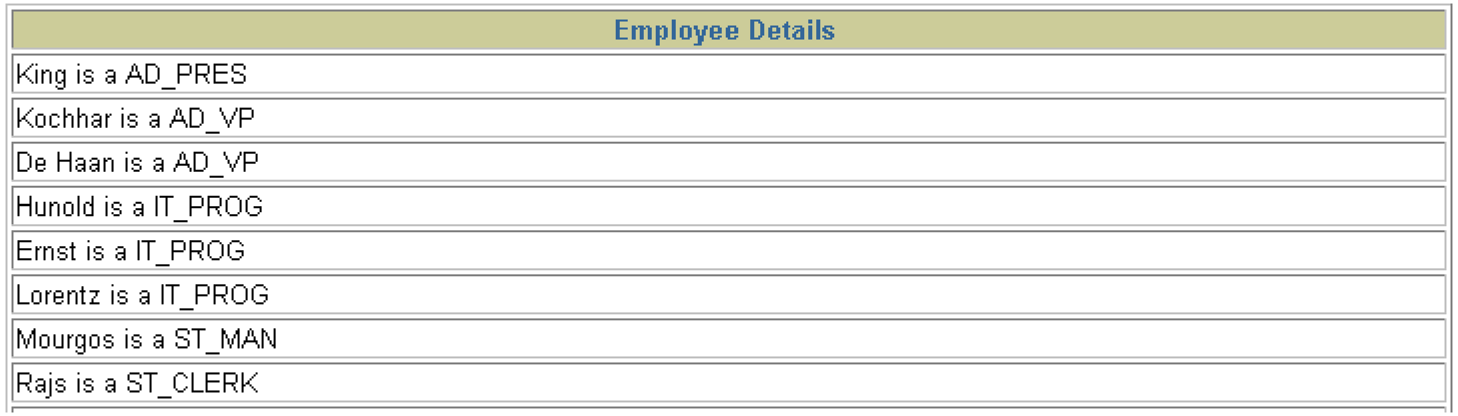
20 rows selected.

Literal Character Strings

- A literal is a character, a number, or a date included in the `SELECT` list.
- Date and character literal values must be enclosed within single quotation marks.
- Each character string is output once for each row returned.

Using Literal Character Strings

```
SELECT last_name  ||' is a '||job_id
       AS "Employee Details"
FROM   employees;
```

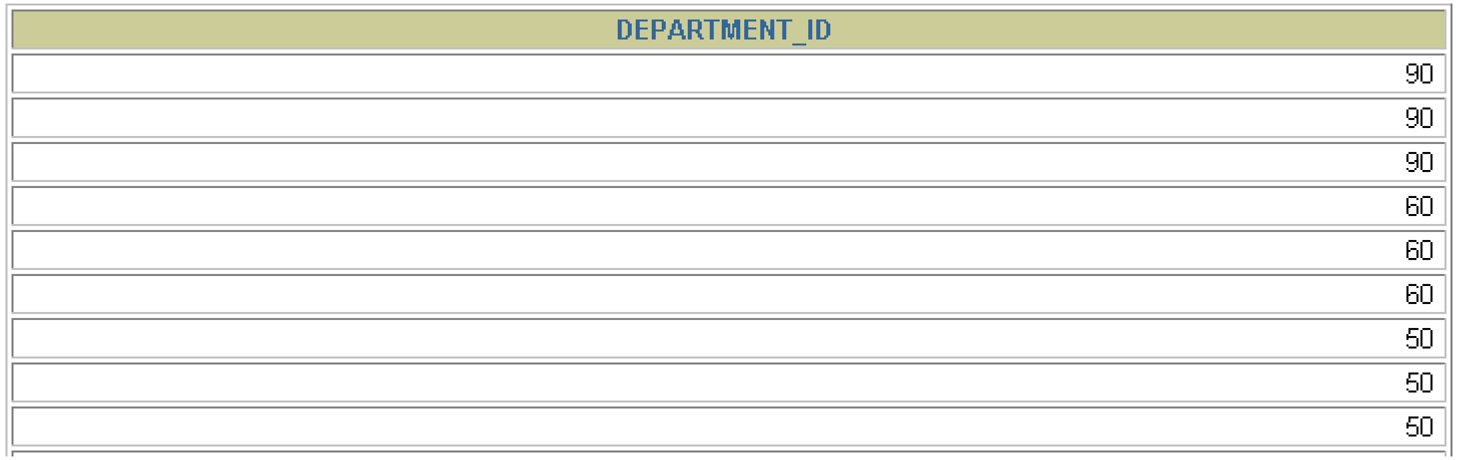


```
Employee Details
King is a AD_PRES
Kochhar is a AD_VP
De Haan is a AD_VP
Hunold is a IT_PROG
Ernst is a IT_PROG
Lorentz is a IT_PROG
Mourgos is a ST_MAN
Rajs is a ST_CLERK
...
20 rows selected.
```

Duplicate Rows

The default display of queries is all rows, including duplicate rows.

```
SELECT department_id
FROM   employees;
```



DEPARTMENT_ID
90
90
90
60
60
60
50
50
50

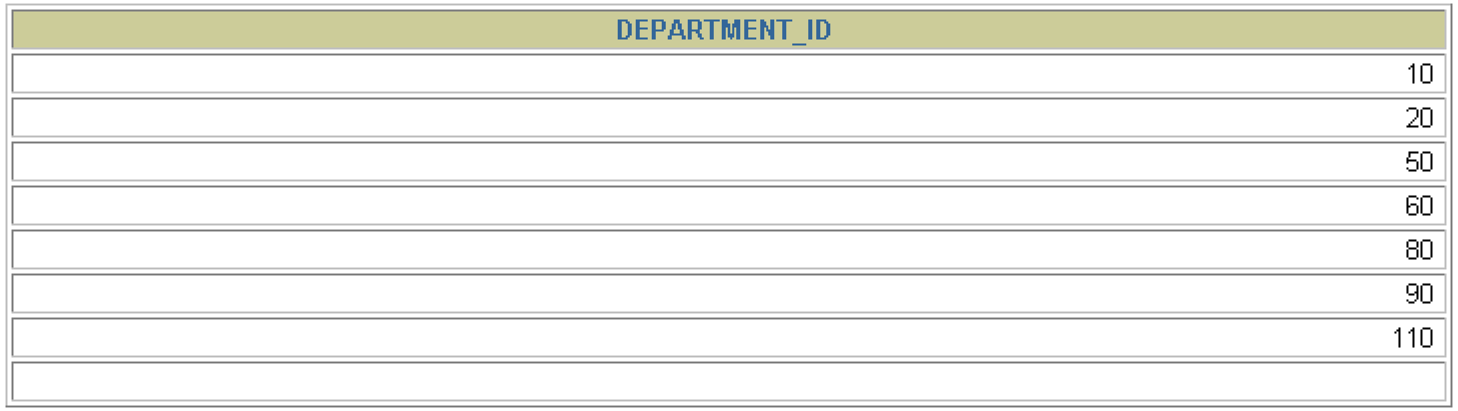
...

20 rows selected.

Eliminating Duplicate Rows

Eliminate duplicate rows by using the `DISTINCT` keyword in the `SELECT` clause.

```
SELECT DISTINCT department_id
FROM   employees;
```



DEPARTMENT_ID
10
20
50
60
80
90
110

8 rows selected.

Limiting Rows Using a Selection

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90
103	Hunold	IT_PROG	60
104	Ernst	IT_PROG	60
107	Lorentz	IT_PROG	60
124	Mourgos	ST_MAN	50

…

20 rows selected.

“retrieve all employees in department 90”

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Limiting the Rows Selected

- Restrict the rows returned by using the `WHERE` clause.

```
SELECT   *|{[DISTINCT] column|expression [alias],...}
FROM     table
[WHERE   condition(s)];
```

- The `WHERE` clause follows the `FROM` clause.

Using the WHERE Clause

```
SELECT employee_id, last_name, job_id, department_id
FROM   employees
WHERE  department_id = 90 ;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Comparison Conditions

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

Using Comparison Conditions

```
SELECT last_name, salary
FROM   employees
WHERE  salary <= 3000;
```

LAST_NAME	SALARY
Matos	2600
Vargas	2500

Other Comparison Conditions

Operator	Meaning
`BETWEEN ...AND...`	Between two values (inclusive),
`IN(set)`	Match any of a list of values
`LIKE`	Match a character pattern
`IS NULL`	Is a null value

Using the BETWEEN Condition

Use the BETWEEN condition to display rows based on a range of values.

```
SELECT last_name, salary
FROM   employees
WHERE  salary BETWEEN 2500 AND 3500;
```

Lower limit **Upper limit**

LAST_NAME	SALARY
Rajs	3500
Davies	3100
Matos	2600
Vargas	2500

Using the `IN` Condition

Use the `IN` membership condition to test for values in a list.

```
SELECT employee_id, last_name, salary, manager_id
FROM   employees
WHERE  manager_id IN (100, 101, 201);
```

EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
202	Fay	6000	201
200	Whalen	4400	101
205	Higgins	12000	101
101	Kochhar	17000	100
102	De Haan	17000	100
124	Mourgos	5800	100
149	Zlotkey	10500	100
201	Hartstein	13000	100

8 rows selected.

Using the `LIKE` Condition

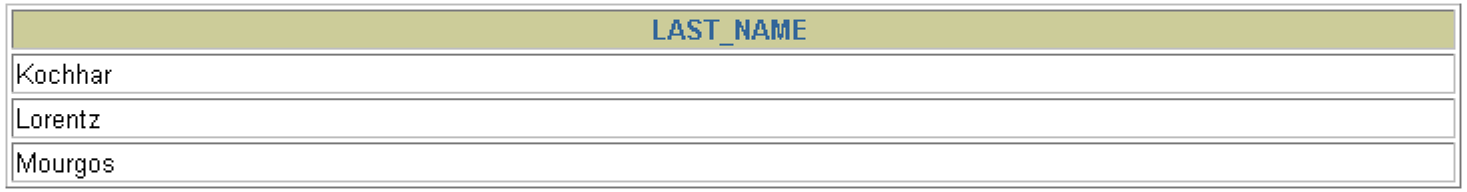
- Use the **`LIKE`** condition to perform wildcard searches of valid search string values.
- Search conditions can contain either literal characters or numbers:
  - `%` denotes zero or many characters.
  - _ denotes one character.

```
SELECT   first_name
FROM     employees
WHERE    first_name LIKE 'S%';
```

Using the `LIKE` Condition

- You can combine pattern-matching characters.

```
SELECT last_name
FROM   employees
WHERE  last_name LIKE '_o%';
```



```
LAST_NAME
Kochhar
Lorentz
Mourgos
```

- You can use the `ESCAPE` identifier to search for the actual % and _ symbols.

Using the `NULL` Conditions

Test for nulls with the `IS NULL` operator.

```
SELECT last_name, manager_id
FROM   employees
WHERE  manager_id IS NULL;
```

LAST_NAME	MANAGER_ID
King	

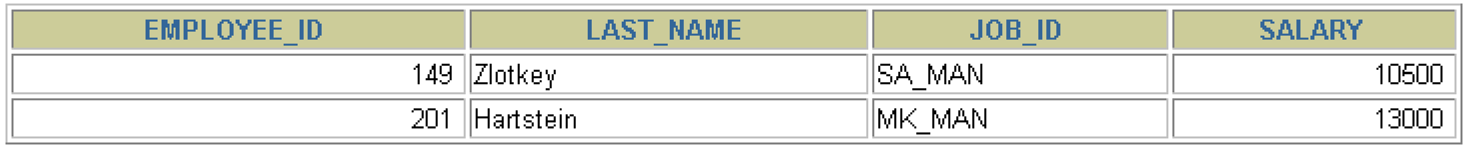
Logical Conditions

Operator	Meaning
`AND`	Returns `TRUE` if *both* component conditions are true
`OR`	Returns `TRUE` if *either* component condition is true
`NOT`	Returns `TRUE` if the following condition is false

Using the AND Operator

AND requires both conditions to be true.

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >=10000
AND    job_id LIKE '%MAN%';
```



EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
149	Zlotkey	SA_MAN	10500
201	Hartstein	MK_MAN	13000

Using the OR Operator

OR requires either condition to be true.

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >= 10000
OR     job_id LIKE '%MAN%';
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
100	King	AD_PRES	24000
101	Kochhar	AD_VP	17000
102	De Haan	AD_VP	17000
124	Mourgos	ST_MAN	5800
149	Zlotkey	SA_MAN	10500
174	Abel	SA_REP	11000
201	Hartstein	MK_MAN	13000
205	Higgins	AC_MGR	12000

8 rows selected.

Using the NOT Operator

```
SELECT last_name, job_id
FROM   employees
WHERE  job_id
       NOT IN ('IT_PROG', 'ST_CLERK', 'SA_REP');
```

LAST_NAME	JOB_ID
King	AD_PRES
Kochhar	AD_VP
De Haan	AD_VP
Mourgos	ST_MAN
Zlotkey	SA_MAN
Whalen	AD_ASST
Hartstein	MK_MAN
Fay	MK_REP
Higgins	AC_MGR
Gietz	AC_ACCOUNT

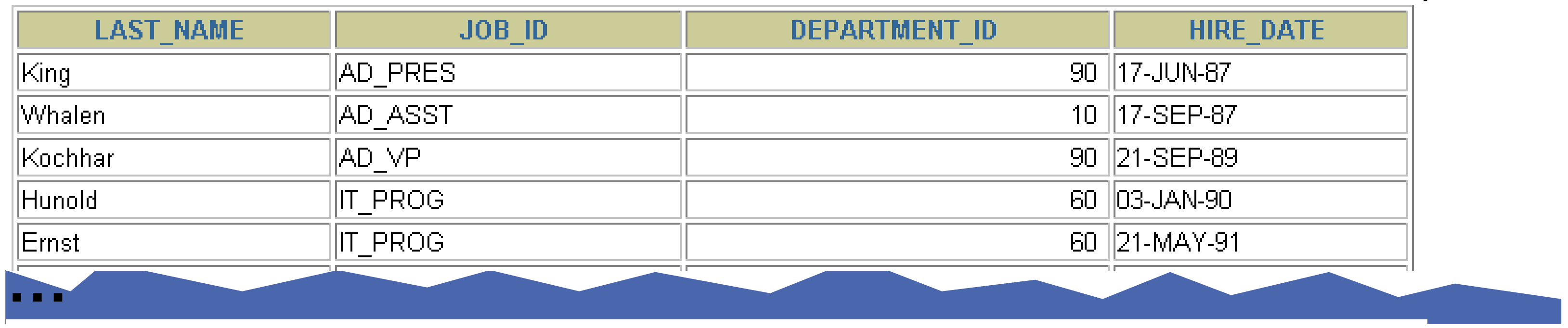
10 rows selected.

ORDER BY Clause

- Sort rows with the ORDER BY clause
  - ASC: ascending order, default
  - DESC: descending order

- The ORDER BY clause comes last in the SELECT statement.

```
SELECT   last_name, job_id, department_id, hire_date
FROM     employees
ORDER BY hire_date ;
```



LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
King	AD_PRES	90	17-JUN-87
Whalen	AD_ASST	10	17-SEP-87
Kochhar	AD_VP	90	21-SEP-89
Hunold	IT_PROG	60	03-JAN-90
Ernst	IT_PROG	60	21-MAY-91

...

20 rows selected.

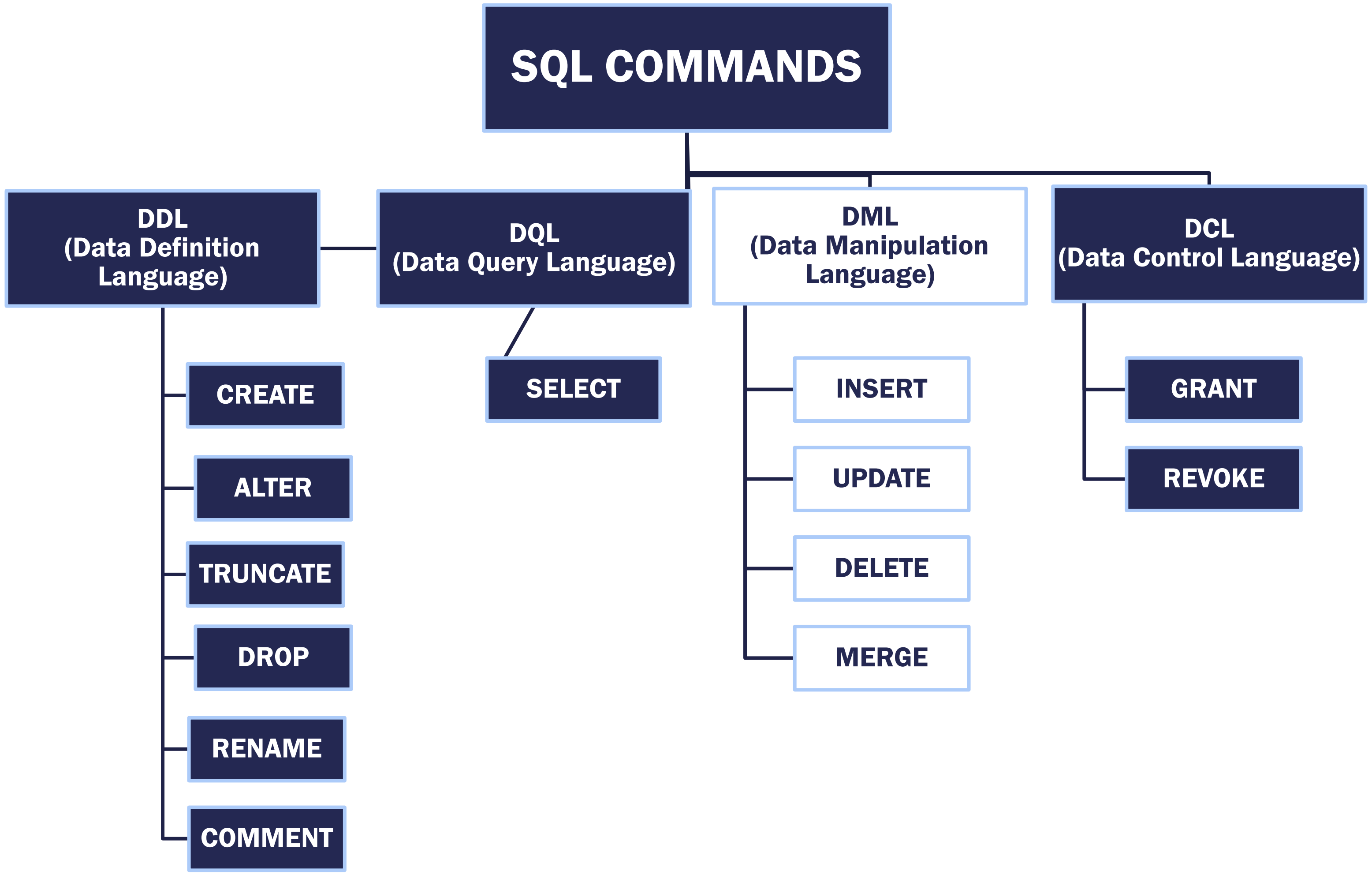
Sorting by Column Alias

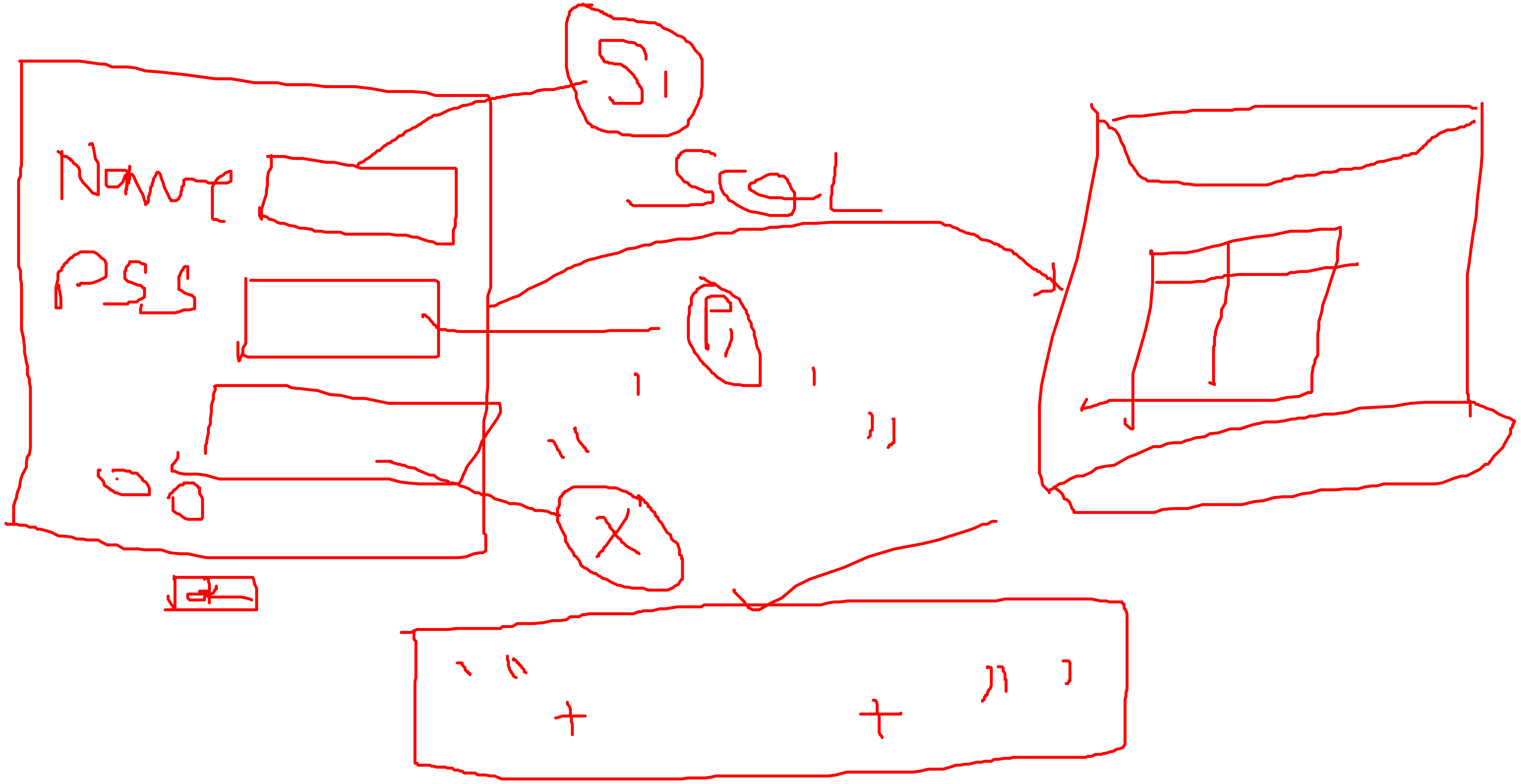
```
SELECT employee_id, last_name, salary*12 annsal
FROM   employees
ORDER BY annsal;
```

EMPLOYEE_ID	LAST_NAME	ANNSAL
144	Vargas	30000
143	Matos	31200
142	Davies	37200
141	Rajs	42000
107	Lorentz	50400
200	Whalen	52800
124	Mourgos	69600
104	Ernst	72000
202	Fay	72000
178	Grant	84000

…

20 rows selected.


SQL COMMANDS
DDL
(Data Definition
Language)
DQL
(Data Query Language)
DML
(Data Manipulation
Language)
DCL
(Data Control Language)
CREATE
ALTER
TRUNCATE
DROP
RENAME
COMMENT
SELECT
INSERT
UPDATE
DELETE
MERGE
GRANT
REVOKE



Name
Pss
ok
S
SQL
P
X
" "
+
+
" "

Data Manipulation Language

- A DML statement is executed when you:
  - Add new rows to a table
  - Modify existing rows in a table
  - Remove existing rows from a table
- A *transaction* consists of a collection of DML statements that form a logical unit of work.

The `INSERT` Statement Syntax

- Add new rows to a table by using the **`INSERT`** statement.

```
INSERT INTO   table [(column [, column...])]
VALUES     (value [, value...]);
```

- Only one row is inserted at a time with this syntax.

Inserting New Rows

- **Insert a new row containing values for each column.**
- **List values in the default order of the columns in the table.**
- **Optionally, list the columns in the `INSERT` clause.**

```
INSERT INTO departments(department_id, department_name,
                        manager_id, location_id)
VALUES      (70, %100, 1700);
1 row created.
```

- **Enclose character and date values within single quotation marks.**

Inserting Rows with Null Values

- Implicit method: Omit the column from the column list.

```
INSERT INTO   departments (department_id,
                           department_name  )
VALUES     (30, 'Purchasing');
1 row created.
```

- Explicit method: Specify the `NULL` keyword in the `VALUES` clause.

```
INSERT INTO   departments
VALUES        (100, 'Finance', NULL, NULL);
1 row created.
```

The UPDATE Statement Syntax

- Modify existing rows with the UPDATE statement.

```
UPDATE     table
SET        column = value [, column = value, ...]
[WHERE     condition];
```

- Update more than one row at a time, if required.

Updating Rows in a Table

- Specific row or rows are modified if you specify the `WHERE` clause.

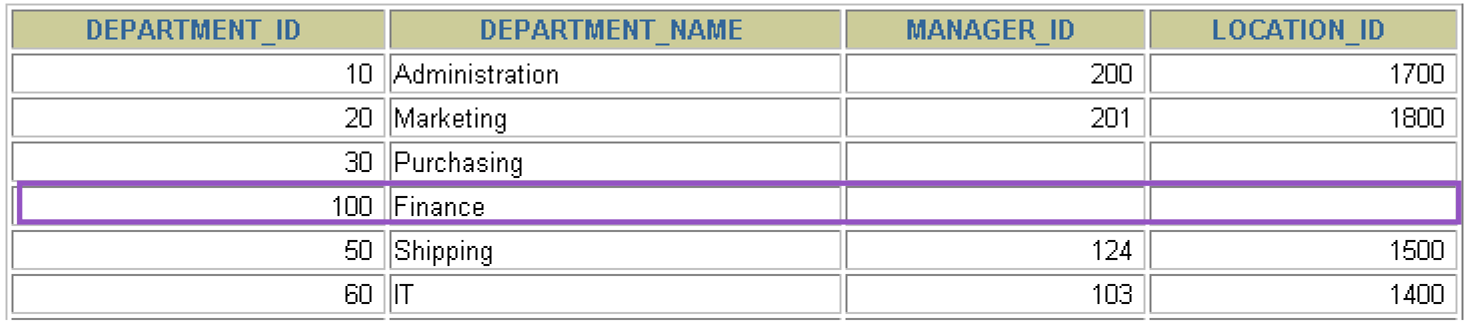
```
UPDATE employees
SET    department_id = 70
WHERE  employee_id = 113;
1 row updated.
```

- All rows in the table are modified if you omit the `WHERE` clause.

```
UPDATE   copy_emp
SET      department_id = 110;
22 rows updated.
```

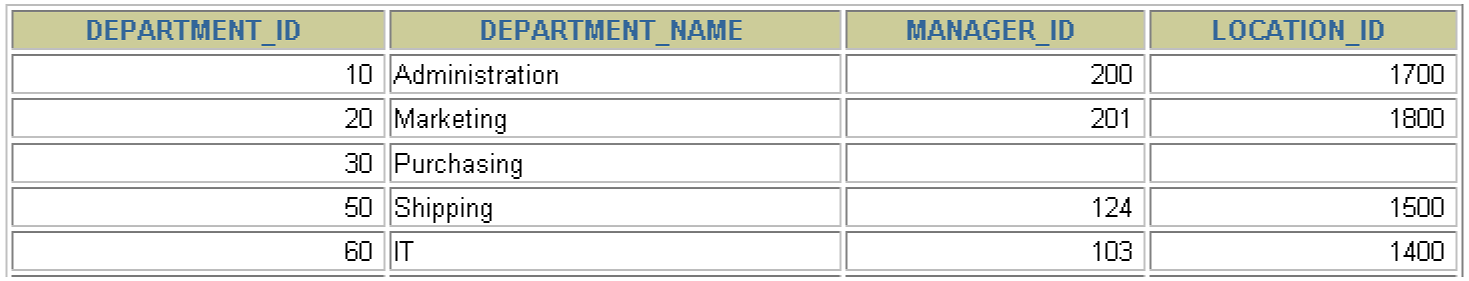
Removing a Row from a Table

DEPARTMENTS



DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing		
100	Finance		
50	Shipping	124	1500
60	IT	103	1400

Delete a row from the DEPARTMENTS table.



DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing		
50	Shipping	124	1500
60	IT	103	1400

The `DELETE` Statement

You can remove existing rows from a table by using the `DELETE` statement.

```
DELETE [FROM]   table
[WHERE          condition];
```

Deleting Rows from a Table

- Specific rows are deleted if you specify the **`WHERE`** clause.

```
 DELETE FROM departments
 WHERE  department_name = 'Finance';
1 row deleted.
```

- All rows in the table are deleted if you omit the **`WHERE`** clause.

```
DELETE FROM  copy_emp;
22 rows deleted.
```

The `MERGE` Statement

- Provides the ability to conditionally update or insert data into a database table
- Performs an **`UPDATE`** if the row exists, and an **`INSERT`** if it is a new row:
  - Avoids separate updates
  - Increases performance and ease of use
  - Is useful in data warehousing applications

The MERGE Statement Syntax

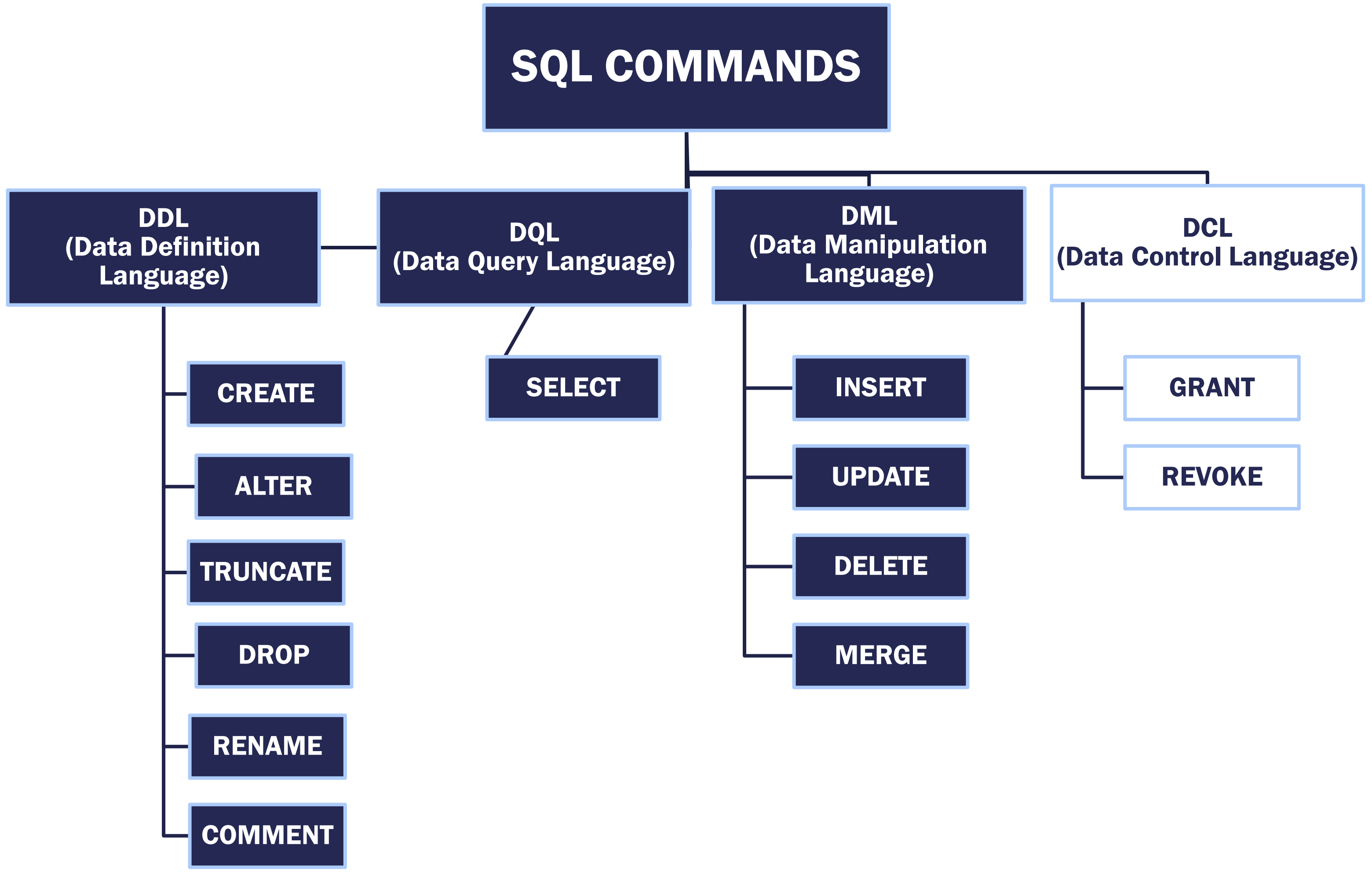
You can conditionally insert or update rows in a table by using the MERGE statement.

```
MERGE INTO table_name table_alias
  USING (table|view|sub_query) alias
  ON (join condition)
  WHEN MATCHED THEN
    UPDATE SET
    col1 = col_val1,
    col2 = col2_val
  WHEN NOT MATCHED THEN
    INSERT (column_list)
    VALUES (column_values);
```

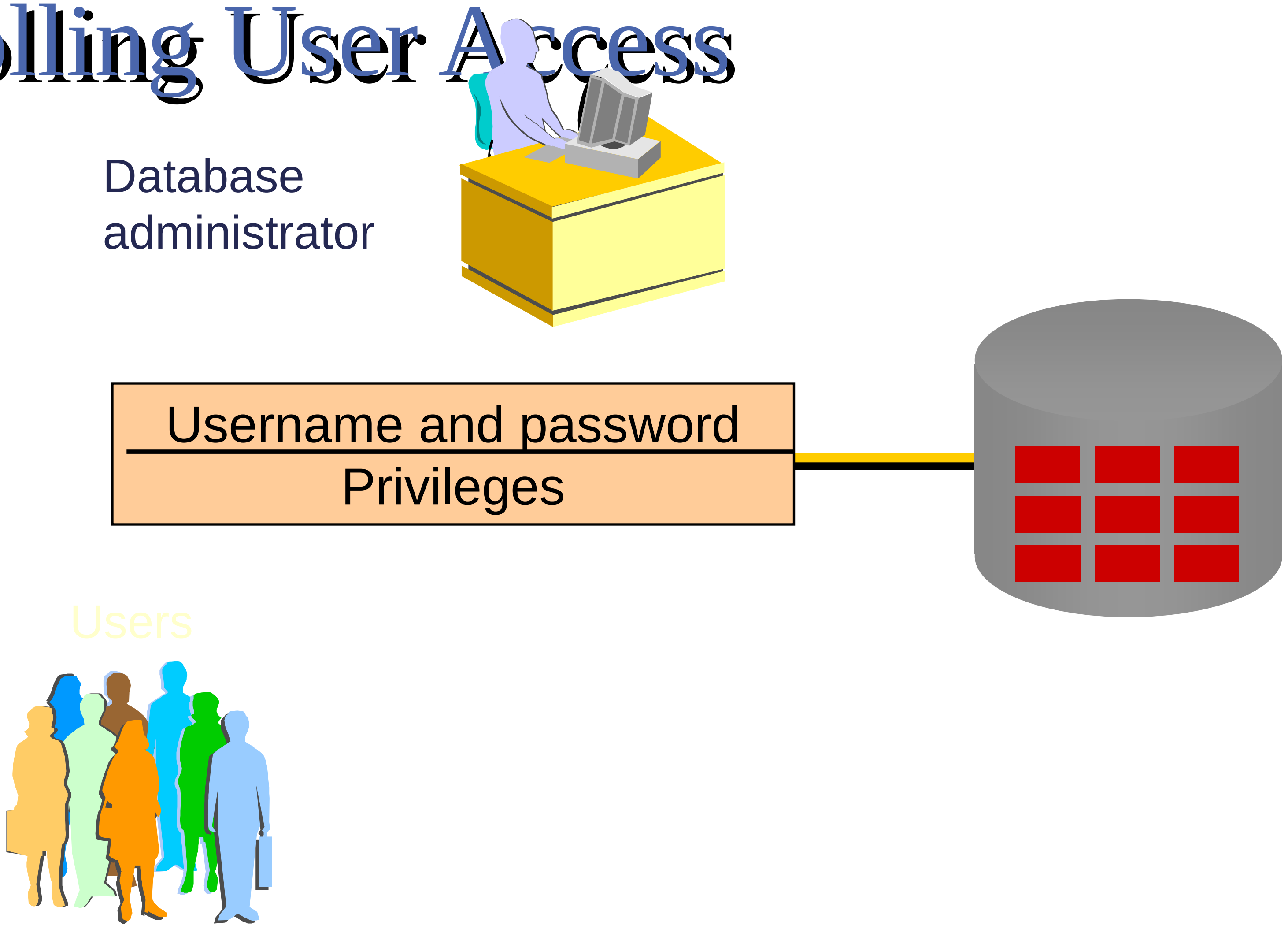
Merging Rows

Insert or update rows in the `COPY_EMP` table to match the `EMPLOYEES` table.

```
MERGE INTO copy_emp  c
  USING employees e
  ON (c.employee_id = e.employee_id)
WHEN MATCHED THEN
  UPDATE SET
     c.first_name     = e.first_name,
     c.last_name      = e.last_name,
     ...
     c.department_id  = e.department_id
WHEN NOT MATCHED THEN
 INSERT VALUES(e.employee_id, e.first_name, e.last_name,
          e.email, e.phone_number, e.hire_date, e.job_id,
          e.salary, e.commission_pct, e.manager_id,
          e.department_id);
```


SQL COMMANDS
DDL (Data Definition Language)
DQL (Data Query Language)
DML (Data Manipulation Language)
DCL (Data Control Language)
CREATE
ALTER
TRUNCATE
DROP
RENAME
COMMENT
SELECT
INSERT
UPDATE
DELETE
MERGE
GRANT
REVOKE

Controlling User Access



Privileges

- Database security:
  - System security
  - Data security
- System privileges: Gaining access to the database
- Object privileges: Manipulating the content of the database objects
- Schemas: Collections of objects, such as tables, views, and sequences

Creating Users

The DBA creates users by using the `CREATE USER` statement.

```
CREATE USER user
IDENTIFIED BY   password;
```

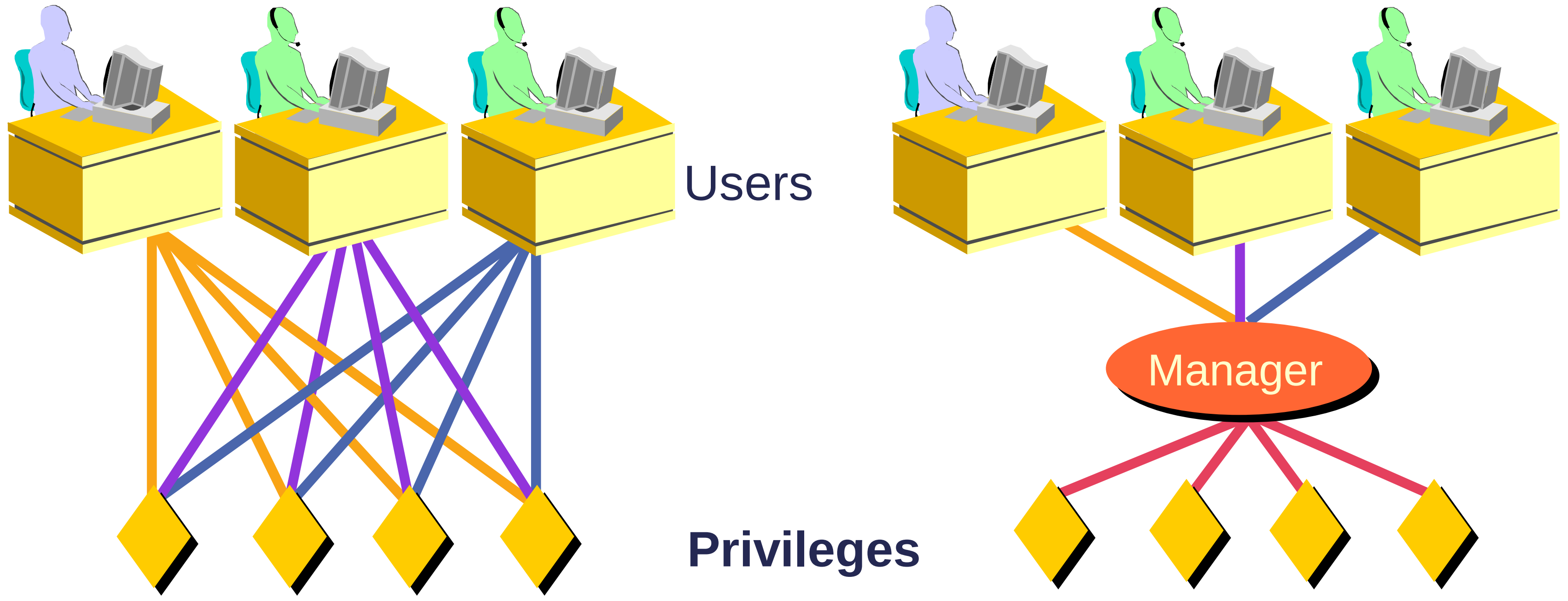
```
CREATE USER  scott
IDENTIFIED BY   tiger;
User created.
```

Granting System Privileges

The DBA can grant a user specific system privileges.

```
GRANT  create session, create table,
       create sequence, create view
TO     scott;
Grant succeeded.
```

What is a Role?



Allocating privileges without a role

Allocating privileges with a role

Creating and Granting Privileges to a Role

- Create a role

```
CREATE ROLE manager;
Role created.
```

- Grant privileges to a role

```
GRANT create table, create view
TO manager;
Grant succeeded.
```

- Grant a role to users

```
GRANT manager TO DEHAAN, KOCHHAR;
Grant succeeded.
```

Granting Object Privileges

- **Grant query privileges on the `EMPLOYEES` table.**

```
GRANT  select
ON     employees
TO     sue, rich;
Grant succeeded.
```

- **Grant privileges to update specific columns to users and roles.**

```
GRANT  update (department_name, location_id)
ON     departments
TO     scott, manager;
Grant succeeded.
```

Using the `WITH GRANT OPTION` and `PUBLIC` Keywords

- **Give a user authority to pass along privileges.**

```
GRANT  select, insert
ON     departments
TO     scott
WITH   GRANT OPTION;
Grant succeeded.
```

- **Allow all users on the system to query data from Alice's `DEPARTMENTS` table.**

```
GRANT  select
ON     alice.departments
TO     PUBLIC;
Grant succeeded.
```

How to Revoke Object Privileges

- **You use the `REVOKE` statement to revoke privileges granted to other users.**
- **Privileges granted to others through the `WITH GRANT OPTION` clause are also revoked.**

```
REVOKE {privilege [, privilege...]|ALL}
ON     object
FROM   {user[, user...]|role|PUBLIC}
[CASCADE CONSTRAINTS];
```

Revoking Object Privileges

As user Alice, revoke the `SELECT` and `INSERT` privileges given to user Scott on the `DEPARTMENTS` table.

```
REVOKE  select, insert
ON      departments
FROM    scott;
Revoke succeeded.
```

THANK YOU!