DATABASE SYSTEMS PROGRAMMING

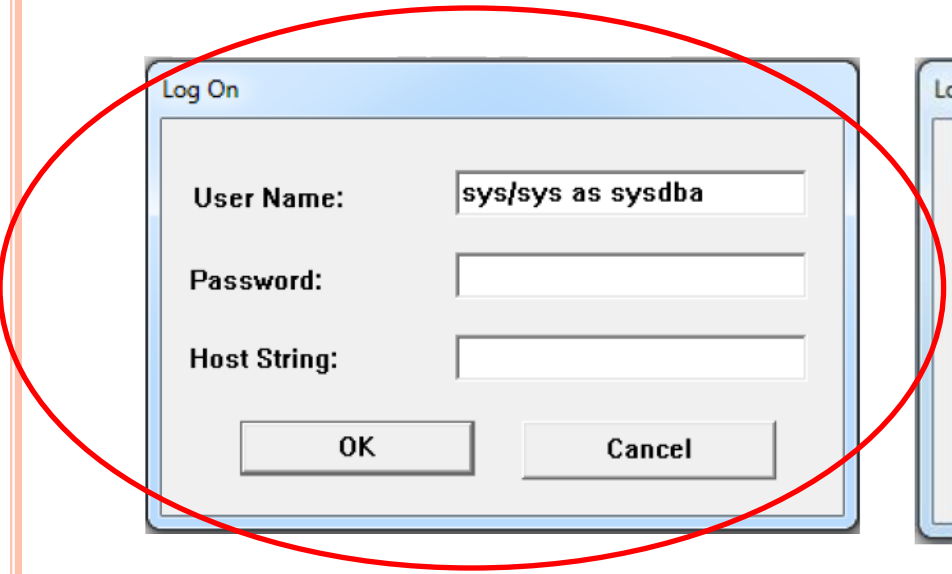
ORACLE®

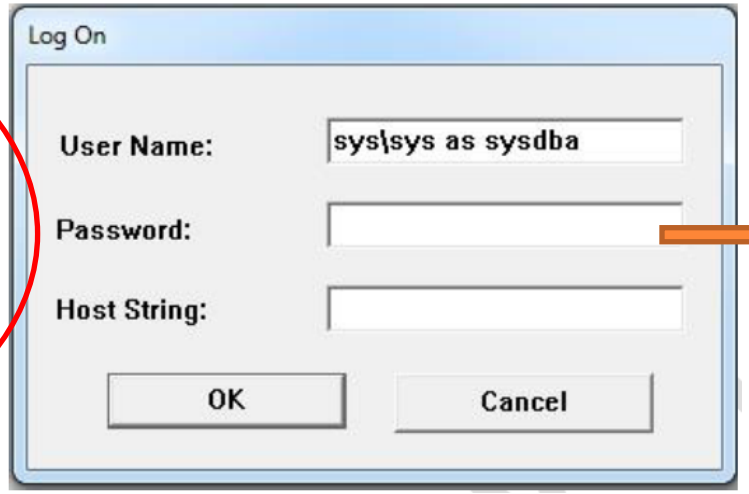
Dr. Katrina Sundus
Computer Science Department – Al-Zaytoonah University 2024-2025

Outlines

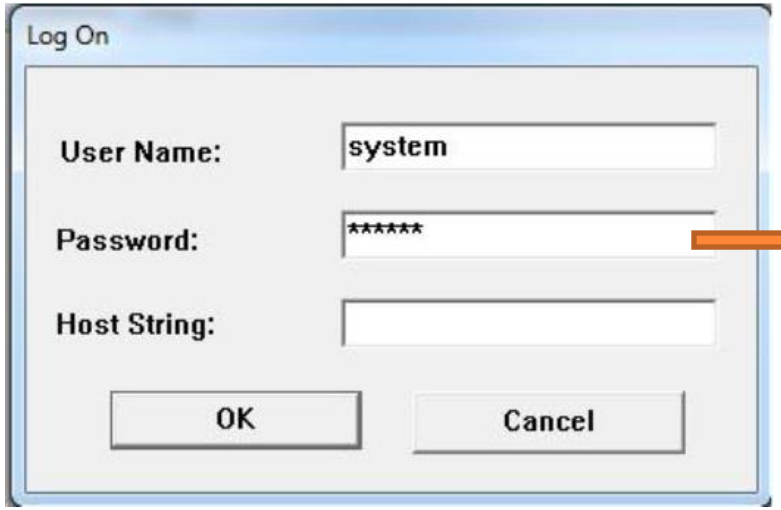
- Connecting to the database
  - As a system administrator
  - As a system user.
- Data Definition Language (DDL) Statements
- Creating Constraints
- Data Manipulation Language (DML) Statements

Connecting To The Database As A System Administrator





Password: sys



system

Connecting To The Database As A System Administrator

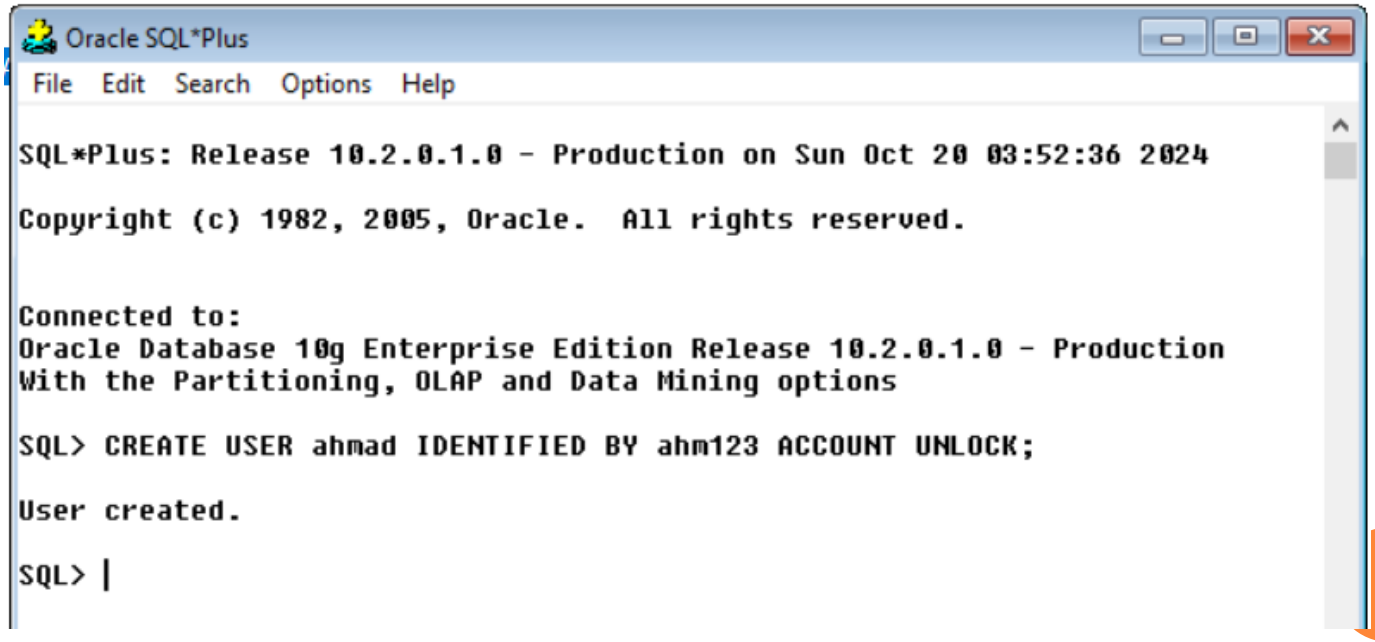
1. CREATE USER command

username: The name of the new user you want to create.
password: The password that the user will use to log in to the database.

- **Syntax**

CREATE USER <username> IDENTIFIED BY <password> ACCOUNT UNLOCK;

- Example:



Connecting To The Database As A System Administrator

2. Grant Command: Give user privilege / role to access the database

- **Syntax**

GRANT CONNECT, RESOURCE, CREATE TABLE TO <username>;

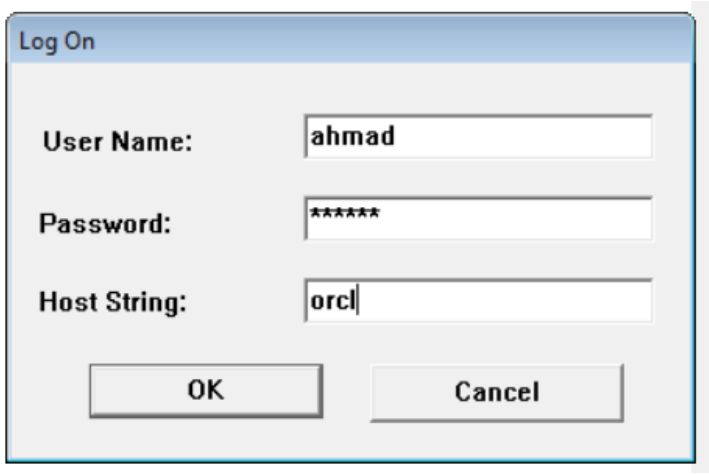
- Example:

```
SQL> grant connect, resource, create table to ahmad;

Grant succeeded.

SQL>
```

- The **Grant statement** allows the specified user to:
  - Connect to the database (CONNECT privilege).
  - Create and manage various database objects (RESOURCE role).
  - Specifically create tables in their schema (CREATE TABLE privilege).

Connecting To The Database As A System User

- Connect to the oracle <username> database




```
Connected to:
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL>
```

DDL Statements summary

Function	Description
CREATE	• create new database objects like tables, views, indexes, or schemas.
ALTER	• Used to modify the structure of an existing database object. • Common use cases include adding, modifying, or dropping columns in a table.
DROP	• Used to delete database objects like tables, views, or indexes. • Once a table is dropped, all data and structure are permanently removed.
TRUNCATE	• Used to remove all rows from a table quickly without deleting the table itself. • It is faster than the DELETE statement because it does not generate individual row deletions in the log.
RENAME	• Used to rename an existing database object.
COMMENT	• Adds a comment to the data dictionary to describe a database object. • These comments can be helpful for documentation and understanding the purpose of tables and columns in the database.

Key Characteristics of DDL Statements

- **Automatically Commit Changes:** DDL statements commit the current transaction, meaning changes are immediately saved to the database.
- **Affect Database Structure:** They change the structure, not the data itself.
- **Irreversible Operations:** Some DDL operations, like DROP and TRUNCATE, are irreversible, and data cannot be recovered without backups.

General DDL syntax

Use the ALTER TABLE statement to:

- Add a new column
- Modify an existing column
- Define a default value for the new column
- Drop a column

DDL stat.	Syntax
CREATE	CREATE TABLE <tablename> (column1_name data_type(size) , column2_name data_type(size)); … CONSTRAINT const_name PRIMARY KEY (column_name), …
ALTER	ALTER TABLE empst ADD (department_id NUMBER);
DROP	DROP TABLE <tablename>
TRUNCATE	TRUNCATE TABLE <tablename>
RENAME	RENAME TABLE <old> to <new>
COMMENT	COMMENT ON TABLE <tablename> IS 'comment'; SELECT * FROM user_tab_comments; // to retrieve comment

PROJECT

EMP Table Description

PK EMPNO NUMBER (4)
Not Null ENAME VARCHAR2 (10)
Check JOB VARCHAR2 (9)
MGR NUMBER (4)
HIREDATE DATE
SAL NUMBER (7, 2)
COMM NUMBER (7, 2)
FK DEPTNO NUMBER (2)

DEPT Table Description

PK DEPTNO NUMBER (2)
Not Null DNAME VARCHAR2 (20)
Not Null LOC VARCHAR2 (20)

SALGRADE Table Description

PK GRADE NUMBER
Not Null LOSAL NUMBER
Not Null HISAL NUMBER

- Which table we should start with?
  Tables with no FK
- Lets talk about constraints then back to our project.

Including Constraints

- Constraints enforce rules at the table level.
- Constraints prevent the deletion of a table if there are dependencies.
- The following constraint types are valid:
  - NOT NULL
  - UNIQUE
  - PRIMARY KEY
  - FOREIGN KEY
  - CHECK

Defining Constraints

- **Syntax:**

```
CREATE TABLE [schema.]table
      (column datatype [DEFAULT expr]
      [column_constraint],
      ...
      [table_constraint][,...]);
```

- **Column-level constraint:**

```
column [CONSTRAINT constraint_name] constraint_type,
```

- **Table-level constraint:**

```
column,...
  [CONSTRAINT constraint_name] constraint_type
  (column, ...),
```

Defining Constraints

Example:

- **Column-level constraint:**

```
CREATE TABLE employees(
  employee_id  NUMBER(6)
    CONSTRAINT emp_emp_id_pk PRIMARY KEY,
  first_name   VARCHAR2(20),
  ...);
```

1

- **Table-level constraint:**

```
CREATE TABLE employees(
  employee_id  NUMBER(6),
  first_name   VARCHAR2(20),
  ...
  job_id       VARCHAR2(10) NOT NULL,
  CONSTRAINT emp_emp_id_pk
    PRIMARY KEY (EMPLOYEE_ID));
```

2

General constraints syntax

constraints	Syntax
NOT NULL	Feature_name Type(size) **constraint** <Tablename_featurename_NN> **not null,**
PRIMARY KEY	**CONSTRAINT** <Tablename_featurename_PK> **PRIMARY KEY** (Feature_name)
FOREIGN KEY	**CONSTRAINT** <Tablename_featurename_FK> **FOREIGN KEY** (Feature_name) **REFERENCES** Tablename(Feature_name)
CHECK	Feature_name Type(size) **constraint** <Tablename_featurename_FK> **CHECK** ((Feature_name = 'sth.1') **OR** (Feature_name = 'sth.2') **OR** … (Feature_name = 'sth.n'))
UNIQUE	**CONSTRAINT** <Tablename_featurename_UK> **UNIQUE**(Featurename)

Example:

DEPT Table Description

PK DEPTNO NUMBER (2)
Not Null DNAME VARCHAR2 (20)
Not Null LOC VARCHAR2 (20)

```
SQL> create table DEPAT(
  2  DEPATNO number(2),
  3  DEPATNAME VARCHAR2(20) constraint DDEPAT_DEPATNAME_NN not null,
  4  LOCATION varchar2(20) constraint DEPAT_LOCATION_NN NOT NULL,
  5  CONSTRAINT DEPAT_DEPATNO_PK PRIMARY KEY (DEPATNO)
  6  );

Table created.
```

SALGRADE Table Description

PK GRADE NUMBER
Not Null LOSAL NUMBER
Not Null HISAL NUMBER

```
SQL> create table SALGRADE(
  2  GRADE NUMBER (1),
  3  LOSAL NUMBER (4) constraint SALGRADE_LOSAL_NN NOT NULL,
  4  HISAL NUMBER (4) constraint SALGRADE_HISAL_NN NOT NULL,
  5  CONSTRAINT SALGRADE_GRADE_PK PRIMARY KEY (GRADE)
  6  );

Table created.
```

EMP Table Description

PK EMPNO NUMBER (4)
Not Null ENAME VARCHAR2 (10)
Check JOB VARCHAR2 (9)
MGR NUMBER (4)
HIREDATE DATE
SAL NUMBER (7, 2)
COMM NUMBER (7, 2)
FK DEPTNO NUMBER (2)

```
SQL> create table EMPLOYEE(
  2  EMPLOYEENO NUMBER(4),
  3  ENAME VARCHAR2(10) constraint EMPLOYEE_ENAME_NN NOT NULL,
  4  JOB VARCHAR2(9) constraint EMPLOYEE_JOB_CC CHECK ((JOB = 'CLERK') OR
  5  (JOB = 'SALESMAN') OR (JOB = 'ANALYST') OR (JOB = 'MANAGER') OR (JOB = 'PRESEDENT') OR
  6  (JOB = 'ACCOUNTANT')),
  7  MGR NUMBER(4),
  8  HIRE_DATE DATE,
  9  SAL NUMBER(7,2),
 10  COMM NUMBER(7,2),
 11  DEPATNO number(2),
 12  CONSTRAINT EMPLOYEE_EMPLOYEENO_PK PRIMARY KEY (EMPLOYEENO),
 13  CONSTRAINT EMPLOYEE_DEPATNO_FK FOREIGN KEY (DEPATNO) REFERENCES DEPAT(DEPATNO)
 14  );

Table created.
```

ALTER

	ALTER	Example
1	Adding a Constraint to an Existing Table	ALTER TABLE EMPLOYEE ADD CONSTRAINT EMPLOYEE_email_UQ UNIQUE (email);
2	Dropping an Existing Constraint	ALTER TABLE EMPLOYEE DROP CONSTRAINT EMPLOYEE_email_UQ;
3	Add a New Column to a Table	ALTER TABLE EMPLOYEE ADD Section_id NUMBER DEFAULT 1 NOT NULL;
4	Modify an Existing Column (TYPE, CONSTRAINT)	ALTER TABLE EMPLOYEE MODIFY ENAME VARCHAR2(100);
5	Drop a Column	ALTER TABLE EMPLOYEE DROP COLUMN Section_id;
6	Rename a Column	ALTER TABLE EMPLOYEE RENAME COLUMN ENAME TO surname;
7	Rename a Table	ALTER TABLE EMPLOYEE RENAME TO staff_members;

DML Statements summary

Transaction consists of a collection of DML statements that form a logical unit of work.

DML Statment	Description
INSERT	• Adding a New Row to a Specific Table
SELECT	• Retrieve data from one or more tables.
Update	• Modify existing records in a table.
Delete	• delete records from a table using the DELETE statement.

Commit;

General DDL syntax

DDL stat.	Syntax
INSERT	INSERT INTO <table_name>(feature1, … , feature_n) VALUES (value1, … , value_n);
SELECT	SELECT <feature1, … , feature_n> FROM <table_name>;
UPDATE	UPDATE table_name SET column1 = value1, column2 = value2, ... WHERE condition;
DELETE	DELETE FROM <tablename> WHERE condition;

Example:

SALGRADE Table Description

PK GRADE NUMBER
Not Null LOSAL NUMBER
Not Null HISAL NUMBER

```
SQL> INSERT INTO SALGRADE(GRADE, LOSAL, HISAL) VALUES (1, 0700, 1200);

1 row created.

SQL> INSERT INTO SALGRADE(GRADE, LOSAL, HISAL) VALUES (2, 1201, 1400);

1 row created.

SQL> INSERT INTO SALGRADE(GRADE, LOSAL, HISAL) VALUES (3, 1401, 2000);

1 row created.

SQL> INSERT INTO SALGRADE(GRADE, LOSAL, HISAL) VALUES (4, 2001, 3000);

1 row created.

SQL> INSERT INTO SALGRADE(GRADE, LOSAL, HISAL) VALUES (5, 3001, 9999);

1 row created.

SQL> select * from salgrade;

     GRADE      LOSAL      HISAL
---------- ---------- ----------
         1        700       1200
         2       1201       1400
         3       1401       2000
         4       2001       3000
         5       3001       9999
```

DEPT Table Description

PK DEPTNO NUMBER (2)
Not Null DNAME VARCHAR2 (20)
Not Null LOC VARCHAR2 (20)

```
SQL>
SQL> INSERT INTO DEPAT(DEPATNO, DEPATNAME, LOCATION) VALUES (60, 'HR', 'MUMBAI');

1 row created.

SQL> INSERT INTO DEPAT(DEPATNO, DEPATNAME, LOCATION) VALUES (10, 'ACCOUNTING', 'NEW YORK');

1 row created.

SQL> INSERT INTO DEPAT(DEPATNO, DEPATNAME, LOCATION) VALUES (20, 'RESEARCH', 'DALLAS');

1 row created.

SQL> INSERT INTO DEPAT(DEPATNO, DEPATNAME, LOCATION) VALUES (30, 'SALES', 'CHICAGO');

1 row created.

SQL> INSERT INTO DEPAT(DEPATNO, DEPATNAME, LOCATION) VALUES (40, 'OPERATIONS', 'BOSTON');

1 row created.

SQL> INSERT INTO DEPAT(DEPATNO, DEPATNAME, LOCATION) VALUES (50, 'IT', 'SINGAPUR');

1 row created.

SQL> SELECT * FROM DEPAT;

   DEPATNO DEPATNAME            LOCATION
---------- -------------------- --------------------
        60 HR                   MUMBAI
        10 ACCOUNTING           NEW YORK
        20 RESEARCH             DALLAS
        30 SALES                CHICAGO
        40 OPERATIONS           BOSTON
        50 IT                   SINGAPUR

6 rows selected.
```

EMP Table Description

PK EMPNO NUMBER (4)
Not Null ENAME VARCHAR2 (10)
Check JOB VARCHAR2 (9)
MGR NUMBER (4)
HIREDATE DATE
SAL NUMBER (7, 2)
COMM NUMBER (7, 2)
FK DEPTNO NUMBER (2)

```
SQL> INSERT INTO EMPLOYEE(EMPLOYEENO, ENAME, JOB, MGR, HIRE_DATE, SAL, COMM, DEPATNO) VALUES (1130,
'Louis', 'CLERK', 1212, SYSDATE, 850.23, 101.20, 10);

1 row created.
```

```
EMPLOYEENO ENAME      JOB               MGR HIRE_DATE         SAL        COMM
---------- ---------- --------- ---------- --------- ---------- ----------
   DEPATNO
----------
      1130 Louis      CLERK            1212 24-OCT-24     850.23       101.2
        10
```

SELECT

	Select	Example
1	Basic SELECT	SELECT * FROM <table_name>;
2	Selecting Specific Columns	SELECT <feature1, … , feature_n> FROM <table_name>;
3	WHERE Clause to Filter Rows	SELECT <feature1, … , feature_n> FROM <table_name> WHERE condition;
4	ORDER BY Clause to Sort Results	SELECT <feature1, … , feature_n> FROM <table_name> ORDER BY feature [ASC | DESC];
5	DISTINCT to Remove Duplicates	SELECT DISTINCT <feature> FROM <table_name> ;
6	Using Aggregate Functions	SELECT AVG(feature) FROM <table_name> WHERE feature = N;
7	…	…

SELECT

```
SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
DEPAT                          TABLE
EMPLOYEE                       TABLE
SALGRADE                       TABLE

SQL> |
```

```
SQL> SELECT * FROM SALGRADE;

     GRADE      LOSAL      HISAL
---------- ---------- ----------
         1        700       1200
         2       1201       1400
         3       1401       2000
         4       2001       3000
         5       3001       9999
```

1. Retrieve all tables
2. Select all features and their records
3. Select specific rows
4. Select specific rows with condition.
5. …..

General TRANSACTION CONTROL SYNTAX

TC stat.	Syntax
COMMIT	COMMIT;
ROLL BACK	ROLLBACK;
SAVEPOINT	INSERT INTO <table name> ….; savepoint1; ROLLBACK TO SAVEPOINT savepoint1;

```
SQL> COMMIT;

Commit complete.

SQL>
```

Naming a table using case-sensitive letters.

```
SQL> select * from EMPLOYEESS;
select * from EMPLOYEESS
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> SELECT * FROM employeess;
SELECT * FROM employeess
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> select * from Employeess;
select * from Employeess
              *
ERROR at line 1:
ORA-00942: table or view does not exist


SQL> select * from "Employeess";

no rows selected

SQL>
```

We prefer not to use it!