CSE 311L (Database Management System)

LAB-Week 01

Lab Instructor: Muyeed Ahmed

Objectives:

- Create database tables
- Describe the data types that can be used when specifying column definition
- Table naming rules & Fields Datatypes
- Insert rows into the created table
- Create Department Table
- Execute a basic SELECT statement

Table Naming Rules

Table names and column names:

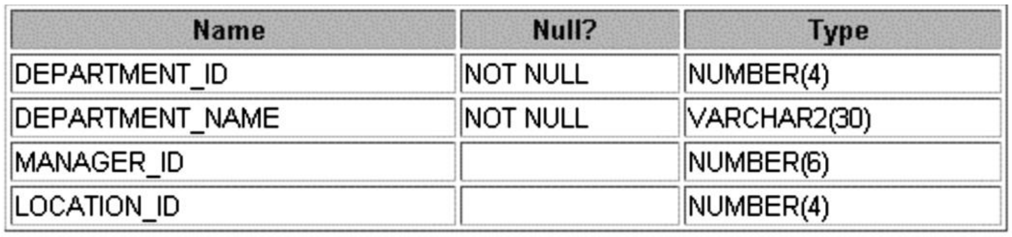
- Must begin with a letter
- Must be 1–30 characters long
- Must contain only A–Z, a–z, 0–9, _, $, and #
- Must not duplicate the name of another object
- owned by the same user
- Must not be an Oracle server reserved word

CREATE TABLE Statement

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

Example

SQL statement for creating the 'Departments' table:



Name	Null?	Type
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

```
CREATE TABLE departments (
        DEPARTMENT_ID NUMBER (4) NOT NULL,
        DEPARTMENT_NAME VARCHAR2 (30) NOT NULL,
        MANAGER_ID NUMBER (6),
        LOCATION_ID NUMBER (4)
);
```

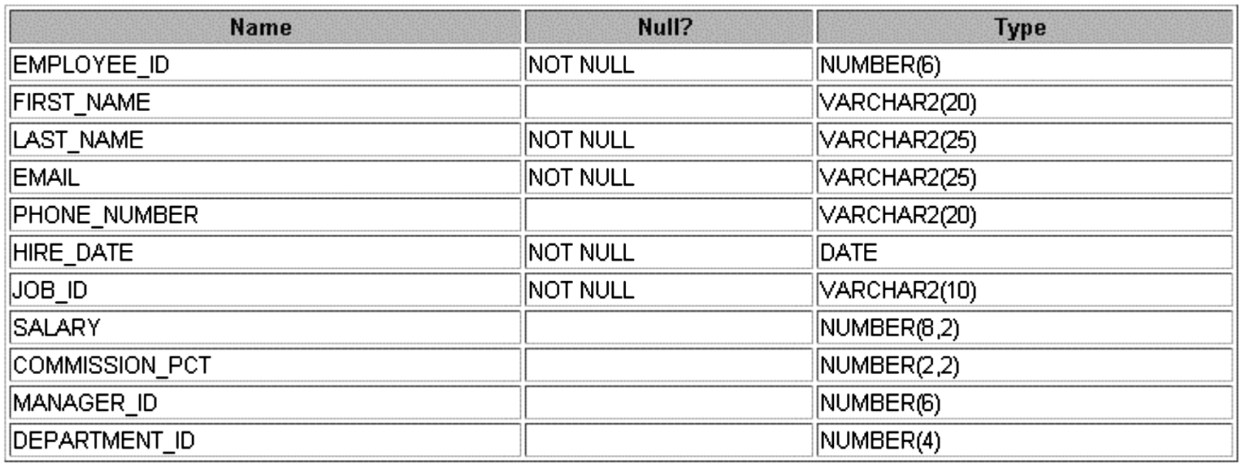
The INSERT Statement Syntax

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

Example

INSERT INTO table (column1, column2, ... column_n)
VALUES (expression1, expression2, ... expression_n);

Activity 01:
Write SQL statement for create the 'Employees' table:



Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

Activity 02:
Write SQL statement for INSERT two employees' data into the employees table you create earlier.

Activity 03:
Write SQL statement for INSERT two Departments' data into the Departments table you just created.

LAB-Week 02

Objective:

- Basic SELECT Statement
- Selecting All Columns, Specific Columns
- Arithmetic Expressions, Using Arithmetic Operators, Parenthesis
- Defining a Column Alias
- Eliminating Duplicate Rows
- Displaying Table Structure
- Concatenation Operator

Basic SELECT Statement

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

Arithmetic Operators

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

Using Column Aliases

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

Using the Concatenation Operator

- Oracle

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

- MySQL

```
SELECT concat (last_name, ' is a ', job_id) "Employee Details"
FROM employees;
```

Eliminating Duplicate Rows

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

Activity 1:

Write a query that displays the last name , weekly salary, department number of the employees. Name the salary column as "Weekly Salary".

Activity 2:

Write a query that displays the last name concatenated with the job ID, separated by a comma and space, and name the column Employee and Title.

LAB-Week 03

Restricting and Sorting Data

- Limiting the Rows Selected
- Restricting with Character Strings and Dates
- Comparison Conditions
- Other Comparison Conditions
- Using the LIKE Condition
- Using the NULL Conditions
- Logical Conditions

Limiting the Rows Selected

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

Character Strings and Dates

```
SELECT last_name, job_id, department_id
FROM employees
WHERE last_name = ‘WHALEN’;
```

Comparison Conditions

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

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

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

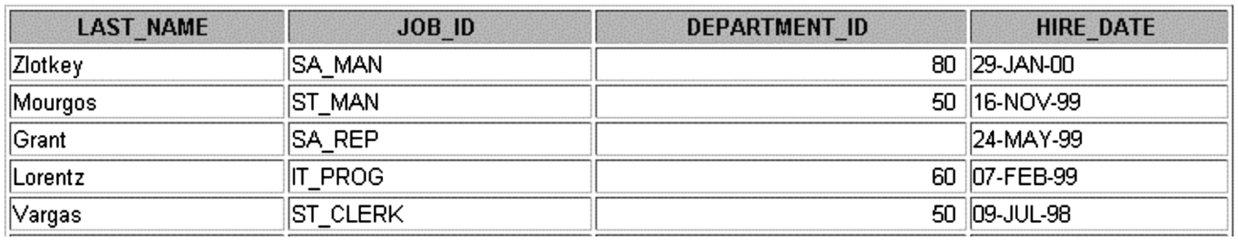
Other Comparison Conditions

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

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

ORDER BY Clause

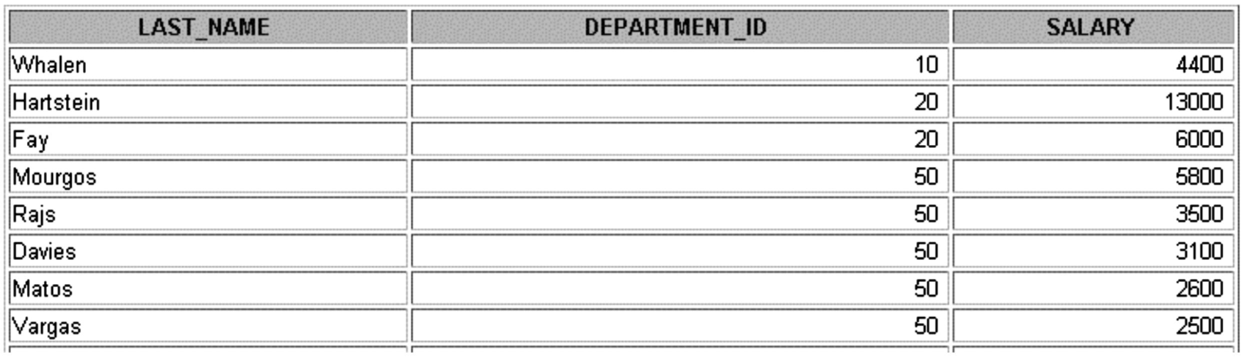
```
SELECT last_name, job_id, department_id, hire_date
FROM employees
ORDER BY hire_date DESC ;
```



LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
Zlotkey	SA_MAN	80	29-JAN-00
Mourgos	ST_MAN	50	16-NOV-99
Grant	SA_REP		24-MAY-99
Lorentz	IT_PROG	60	07-FEB-99
Vargas	ST_CLERK	50	09-JUL-98

Sorting by Multiple Columns

```
SELECT last_name, department_id, salary
FROM employees
ORDER BY department_id, salary DESC;
```



LAST_NAME	DEPARTMENT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000
Mourgos	50	5800
Rajs	50	3500
Davies	50	3100
Matos	50	2600
Vargas	50	2500

Activity 01:
Display the employee last name, job ID, and start date of employees hired between February 20, 1998, and May 1, 1998. Order the query in ascending order by start date.

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.

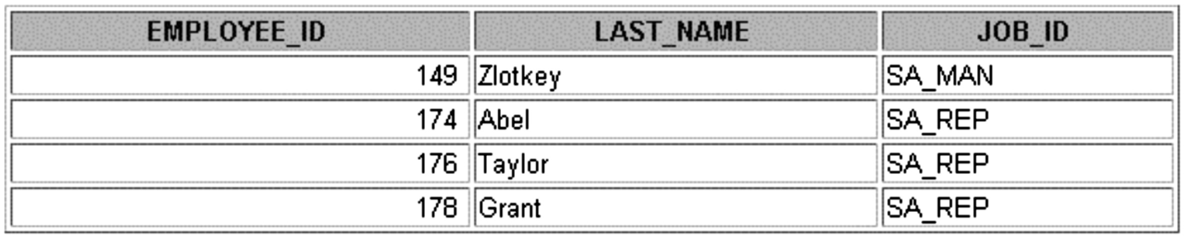
```
SELECT last_name
```

last_name
K**o**chHer
L**o**rentz
M**o**urgos

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

The ESCAPE Option

```
SELECT employee_id, last_name, job_id
FROM employees
WHERE job_id LIKE '%SA\_%';
```

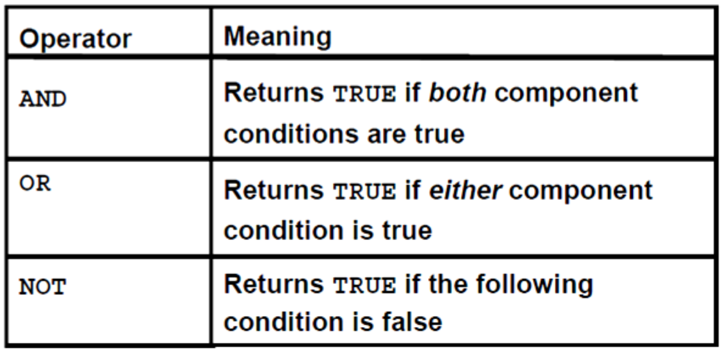


EMPLOYEE_ID	LAST_NAME	JOB_ID
149	Zlotkey	SA_MAN
174	Abel	SA_REP
176	Taylor	SA_REP
178	Grant	SA_REP

Using the NULL Conditions

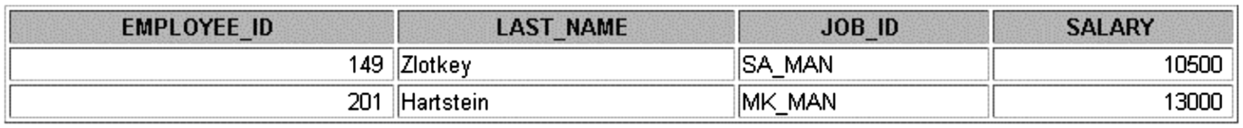
```
SELECT last_name, manager_id
FROM employees
WHERE manager_id IS NULL;
```

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

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

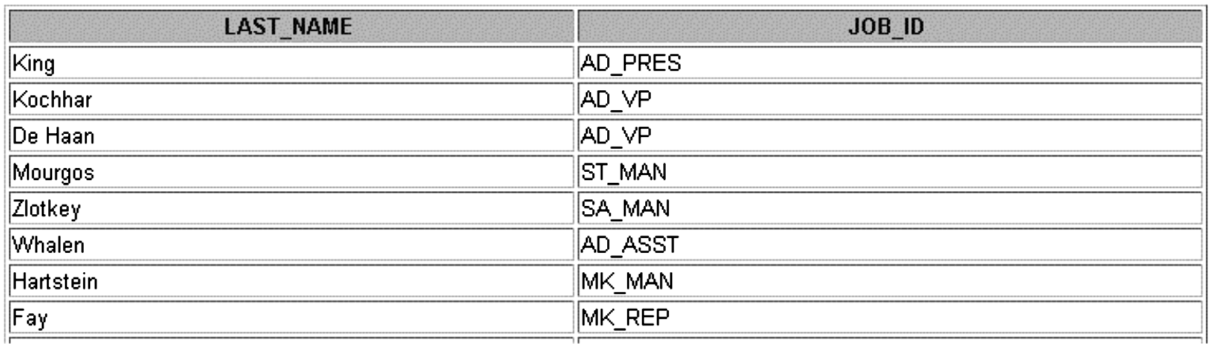


EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
149	Zlotkey	SA_MAN	10500
201	Hartstein	MK_MAN	13000

Using the 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

Activity 01:
Display the last name and hire date of every employee who was hired in 1994.

Activity 02:
Display the last name, salary, and commission for all employees who earn commissions. Sort the data in descending order of salary.