Name: Urooba Gohar
Roll No: 22P-9216
Section: BSCS-6A

Database Systems Lab 3

Task 1:

Using wildcards, perform the following tasks:

1. Get all Employees having ‘A’ anywhere in their names.

Ans:

To get all employees having ‘a’ in their name, we use the “like” operator with “%” wildcard (for multiple characters).

Following is the query:

```
--22P-9216
select * from Employees where first_name like '%a%'
```

Following is the result:

Query Result

SQL | Fetched 50 rows in 0.009 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05	AD_VP	17000	(null)	100	90
2	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG	9000	(null)	102	60
3	105	David	Austin	DAUSTIN	590.423.4569	25-JUN-05	IT_PROG	4800	(null)	103	60
4	106	Valli	Pataballa	VPATABAL	590.423.4560	05-FEB-06	IT_PROG	4800	(null)	103	60
5	107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-07	IT_PROG	4200	(null)	103	60
6	108	Nancy	Greenberg	NGREENBE	515.124.4569	17-AUG-02	FI_MGR	12008	(null)	101	100
7	109	Daniel	Faviet	DFAVIET	515.124.4169	16-AUG-02	FI_ACCOUNT	9000	(null)	108	100
8	111	Ismael	Sciarra	ISCIARRA	515.124.4369	30-SEP-05	FI_ACCOUNT	7700	(null)	108	100
9	112	Jose Manuel	Urman	JMURMAN	515.124.4469	07-MAR-06	FI_ACCOUNT	7800	(null)	108	100
10	115	Alexander	Khoo	AKHOO	515.127.4562	18-MAY-03	PU_CLERK	3100	(null)	114	30
11	117	Sigal	Tobias	STOBIAS	515.127.4564	24-JUL-05	PU_CLERK	2800	(null)	114	30
12	119	Karen	Colmenares	KCOLMENA	515.127.4566	10-AUG-07	PU_CLERK	2500	(null)	114	30
13	120	Matthew	Weiss	MWEISS	650.123.1234	18-JUL-04	ST_MAN	8000	(null)	100	50
14	121	Adam	Fripp	AFRIPP	650.123.2234	10-APR-05	ST_MAN	8200	(null)	100	50
15	122	Payam	Kaufling	PKAUFLIN	650.123.3234	01-MAY-03	ST_MAN	7900	(null)	100	50
16	123	Shanta	Vollman	SVOLLMAN	650.123.4234	10-OCT-05	ST_MAN	6500	(null)	100	50
17	125	Julia	Nayer	JNAYER	650.124.1214	16-JUL-05	ST_CLERK	3200	(null)	120	50

2. Get all Employees having ‘e’ as 2nd last character.

Ans:

To get all employees having ‘e’ as the 2nd last character in their name, we use the “like” operator with “%” wildcard (for multiple characters) and “_” (for a single character).

Following is the query:

```
--22P-9216
select * from Employees where first_name like '%e_'
```

Following is the result:

Query Result

SQL | All Rows Fetched: 33 in 0.008 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	100	Steven	King	SKING	515.123.4567	17-JUN-03	AD_PRES	24000	(null)	(null)	90
2	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01	AD_VP	17000	(null)	100	90
3	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG	9000	(null)	102	60
4	109	Daniel	Faviet	DFAVIET	515.124.4169	16-AUG-02	FI_ACCOUNT	9000	(null)	108	100
5	111	Ismael	Sciarra	ISCIARRA	515.124.4369	30-SEP-05	FI_ACCOUNT	7700	(null)	108	100
6	112	Jose Manuel	Urman	JMURMAN	515.124.4469	07-MAR-06	FI_ACCOUNT	7800	(null)	108	100
7	114	Den	Raphaely	DRAPHEAL	515.127.4561	07-DEC-02	PU_MAN	11000	(null)	100	30
8	115	Alexander	Khoo	AKHOO	515.127.4562	18-MAY-03	PU_CLERK	3100	(null)	114	30
9	119	Karen	Colmenares	KCOLMENA	515.127.4566	10-AUG-07	PU_CLERK	2500	(null)	114	30
10	120	Matthew	Weiss	MWEISS	650.123.1234	18-JUL-04	ST_MAN	8000	(null)	100	50
11	127	James	Landry	JLANDRY	650.124.1334	14-JAN-07	ST_CLERK	2400	(null)	120	50
12	128	Steven	Markle	SMARKLE	650.124.1434	08-MAR-08	ST_CLERK	2200	(null)	120	50
13	131	James	Marlow	JAMRLOW	650.124.7234	16-FEB-05	ST_CLERK	2500	(null)	121	50
14	134	Michael	Rogers	MROGERS	650.127.1834	26-AUG-06	ST_CLERK	2900	(null)	122	50
15	136	Hazel	Philtanker	HPHILTAN	650.127.1634	06-FEB-08	ST_CLERK	2200	(null)	122	50
16	138	Stephen	Stiles	SSTILES	650.121.2034	26-OCT-05	ST_CLERK	3200	(null)	123	50
17	144	Peter	Vargas	PVARGAS	650.121.2004	09-JUL-06	ST_CLERK	2500	(null)	124	50

3. Get all Employees having ‘l’ (small L, not i) as 2nd character.

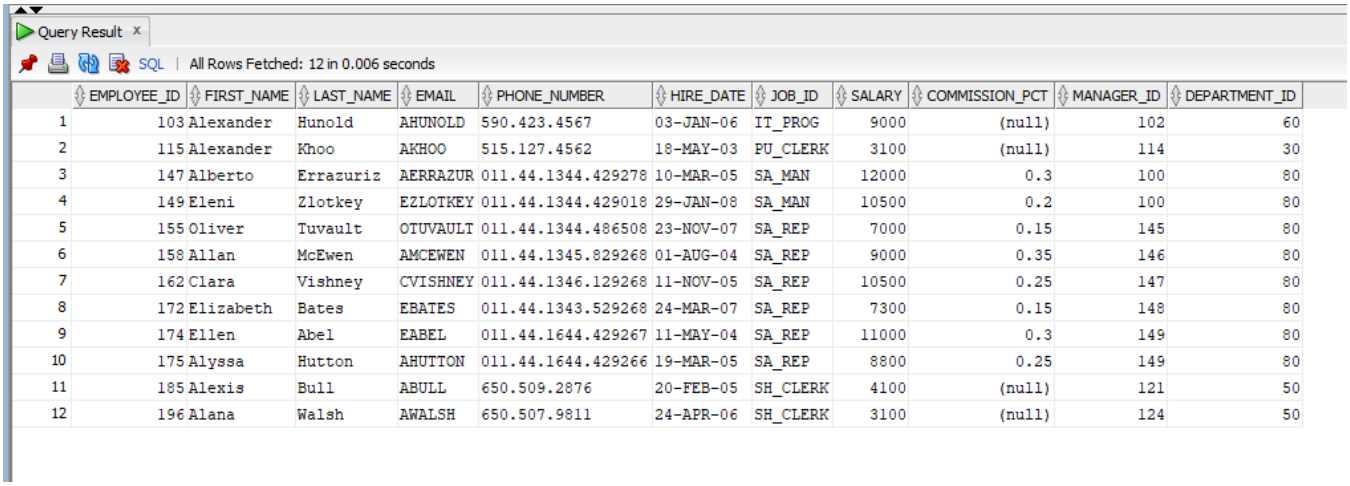
Ans:

To get all employees having ‘l’ as the 2nd character in their name, we use the “like” operator with “%” wildcard (for multiple characters) and “_” (for a single character).

Following is the query:

```
--22P-9216
select * from Employees where first_name like '_l%'
```

Following is the result:



Query Result | SQL | All Rows Fetched: 12 in 0.006 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06	IT_PROG	9000	(null)	102	60
2	115	Alexander	Khoo	AKHOO	515.127.4562	18-MAY-03	PU_CLERK	3100	(null)	114	30
3	147	Alberto	Errazuriz	AERRAZUR	011.44.1344.429278	10-MAR-05	SA_MAN	12000	0.3	100	80
4	149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	29-JAN-08	SA_MAN	10500	0.2	100	80
5	155	Oliver	Tuvault	OTUVAULT	011.44.1344.486508	23-NOV-07	SA_REP	7000	0.15	145	80
6	158	Allan	McEwen	AMCEWEN	011.44.1345.829268	01-AUG-04	SA_REP	9000	0.35	146	80
7	162	Clara	Vishney	CVISHNEY	011.44.1346.129268	11-NOV-05	SA_REP	10500	0.25	147	80
8	172	Elizabeth	Bates	EBATES	011.44.1343.529268	24-MAR-07	SA_REP	7300	0.15	148	80
9	174	Ellen	Abel	EABEL	011.44.1644.429267	11-MAY-04	SA_REP	11000	0.3	149	80
10	175	Alyssa	Hutton	AHUTTON	011.44.1644.429266	19-MAR-05	SA_REP	8800	0.25	149	80
11	185	Alexis	Bull	ABULL	650.509.2876	20-FEB-05	SH_CLERK	4100	(null)	121	50
12	196	Alana	Walsh	AWALSH	650.507.9811	24-APR-06	SH_CLERK	3100	(null)	124	50

4. Get all Employees having ‘l’ as 2nd character and ‘n’ as 4th character.

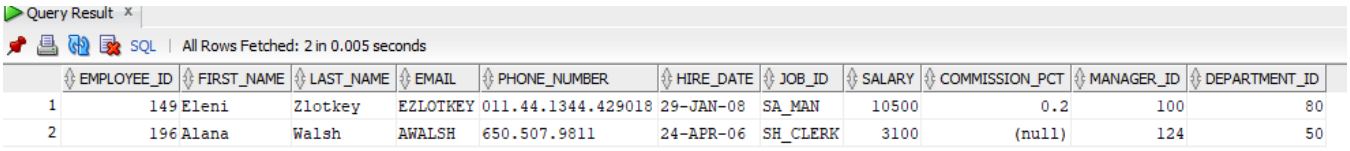
Ans:

To get all employees having ‘l’ as the 2nd character and ‘n’ as the 4th character in their name, we use the “like” operator with “%” wildcard (for multiple characters) and “_” (for a single character).

Following is the query:

```
--22P-9216
select * from Employees where first_name like '_l_n%'
```

Following is the result:



Query Result | SQL | All Rows Fetched: 2 in 0.005 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	29-JAN-08	SA_MAN	10500	0.2	100	80
2	196	Alana	Walsh	AWALSH	650.507.9811	24-APR-06	SH_CLERK	3100	(null)	124	50

Task 2:

Create a new user using SQL command Line and grant privileges. The user should be named after your roll number with lab03 as prefix.

Ans:

To create a new user under the name “P229216_Lab3” and grant all privileges to it, we write the following query:

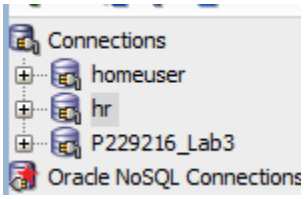
```
--22P-9216
CREATE USER P229216_Lab3 IDENTIFIED BY urooba01
Grant all PRIVILEGES to P229216_Lab3
```

Following is the result:

```
User P229216_LAB3 created.


Grant succeeded.
```

Here we can see that we have three connections now including the one we just created:



Task 3:

Create a table Employees with attributes(columns) named Employee_id, Full_Name, Salary, Department_id, Start_Date, End_Date, Married, Phone_No.

Ans:

To create an Employee table with the given attributes, we use “create table” and give it the name “Employees”.

Following is the query:

```
--22P-9216
CREATE TABLE Employees(
Employee_id NUMBER(5) PRIMARY KEY,
Full_Name VARCHAR2(20) NOT NULL,
Salary NUMBER(6,2) NOT NULL,
Department_id NUMBER(5),
Start_Date DATE NOT NULL,
End_Date DATE,
Married CHAR(1) CHECK (Married IN('Y', 'N')),
Phone_No VARCHAR2(20)
);
```

Following is the result:

```
Table EMPLOYEES created.
```

Task 4:

Create another table Departments with attributes(columns) named Department_id, Department_name, Department_code, Date_Founded.

Ans:

To create a “Department” table with given columns, we run the following query:

```
--22P-9216
CREATE TABLE Departments(
Department_id NUMBER(5) PRIMARY KEY,
Department_Name VARCHAR2(40) UNIQUE NOT NULL,
Department_Code VARCHAR2(40) NOT NULL,
Date_Founded NUMBER(4) CHECK (Date_Founded > 2000)
);
```

Following is the result:

```
Table DEPARTMENTS created.
```

Task 5:

Make sure the department names are unique and check if the date_founded is greater than 2000.

Ans:

The department name is unique since we've added the "unique" keyword with it and the date founded is greater than 2000 by using number check.

Following is the query:

```
--22P-9216
CREATE TABLE Departments(
Department_id NUMBER(5) PRIMARY KEY,
Department_Name VARCHAR2(40) UNIQUE NOT NULL,
Department_Code VARCHAR2(40) NOT NULL,
Date_Founded NUMBER(4) CHECK (Date_Founded > 2000)
);
```

Task 6:

Make sure that you set the IDs in each table to primary keys.

Ans:

The employee_id and department_id are set to primary key.

Following is the query:

```
--22P-9216
CREATE TABLE Employees(
Employee_id NUMBER(5) PRIMARY KEY,
```

```
--22P-9216
CREATE TABLE Departments(
Department_id NUMBER(5) PRIMARY KEY,
```

Task 7:

Make use of alter command to add foreign key constraint and pass reference of departments to the employees table using has-belongs to concept.

Ans:

To add foreign key using alter command, we run the following query:

```
--22P-9216
ALTER TABLE Employees
ADD CONSTRAINT dept_fk FOREIGN KEY (Department_id) REFERENCES Departments(Department_id)
```

Following is the result:

```
Table EMPLOYEES altered.
```

Task 8:

Insert 5 rows of data into both tables.

Ans:

To insert 5 rows of data into the Employees table, we use the “insert into” and “values” commands. Following is the query:

```
--22P-9216
INSERT INTO Employees (Employee_id, Full_Name, Salary, Department_id, Start_Date, End_Date, Married, Phone_No)
VALUES (2001, 'Ali Khan', 7500.50, 1001, TO_DATE('2021-05-12', 'YYYY-MM-DD'), NULL, 'N', '03001234567');

INSERT INTO Employees (Employee_id, Full_Name, Salary, Department_id, Start_Date, End_Date, Married, Phone_No)
VALUES (2002, 'Sara Ahmed', 8500.75, 1002, TO_DATE('2020-08-25', 'YYYY-MM-DD'), NULL, 'Y', '03119876543');

INSERT INTO Employees (Employee_id, Full_Name, Salary, Department_id, Start_Date, End_Date, Married, Phone_No)
VALUES (2003, 'Bilal Saeed', 9000.00, 1003, TO_DATE('2019-02-14', 'YYYY-MM-DD'), NULL, 'N', '03221122334');

INSERT INTO Employees (Employee_id, Full_Name, Salary, Department_id, Start_Date, End_Date, Married, Phone_No)
VALUES (2004, 'Hina Tariq', 7200.25, 1004, TO_DATE('2022-07-01', 'YYYY-MM-DD'), NULL, 'Y', '03334455667');

INSERT INTO Employees (Employee_id, Full_Name, Salary, Department_id, Start_Date, End_Date, Married, Phone_No)
VALUES (2005, 'Usman Tariq', 8800.60, 1005, TO_DATE('2023-01-10', 'YYYY-MM-DD'), NULL, 'Y', '03445566778');
```

Following is the result:

```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.
```

To insert 5 rows of data into the Departments table, we use the “insert into” and “values” commands. Following is the query:

```
--22P-9216
INSERT INTO Departments (Department_id, Department_Name, Department_Code, Date_Founded)
VALUES (1001, 'Software Engineering', 'SE', 2010);

INSERT INTO Departments (Department_id, Department_Name, Department_Code, Date_Founded)
VALUES (1002, 'Human Resources', 'HR', 2005);

INSERT INTO Departments (Department_id, Department_Name, Department_Code, Date_Founded)
VALUES (1003, 'Finance', 'FIN', 2008);

INSERT INTO Departments (Department_id, Department_Name, Department_Code, Date_Founded)
VALUES (1004, 'Marketing', 'MKT', 2012);

INSERT INTO Departments (Department_id, Department_Name, Department_Code, Date_Founded)
VALUES (1005, 'Data Science', 'DS', 2015);
```

Following is the result:

```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.
```

Task 9:

Add Column Speciality in Departments table and set its default value to None.

Ans:

We use “alter” command to add a new column under the name “Speciality” in Departments table. Following is the query:

```
--22P-9216
ALTER TABLE Departments ADD Speciality VARCHAR2(50) DEFAULT 'None';
```

Following is the result:

```
Table DEPARTMENTS altered.
```

Task 10:

Create a table named Jobs with attributes being the same as the table from HR.

Ans:

Here we created a new table under the name “Jobs” having the same attributes as the Job table of HR had.
Following is the query:

```
--22P-9216
CREATE TABLE Jobs(
Job_id INTEGER PRIMARY KEY,
Job_Title VARCHAR2(35) NOT NULL,
Min_Salary NUMBER(6,0),
Max_Salary NUMBER(6,0)
);
```

Following is the result:

```
Table JOBS created.
```

Task 11:

Modify the Job_id to be of Integer Type and make it the primary key.

Ans:

While creating the “Jobs” table, we already set the “Job_id” to integer type while making it a primary key.
Following is the query:

```
--22P-9216
CREATE TABLE Jobs(
Job_id INTEGER PRIMARY KEY,
```

Task 12:

Write a SQL statement to add Employee_id column in jobs table as foreign key referencing to the primary key Employee_id of Employees table.

Ans:

“Alter” command is again used to add a new column “Employee_id” to the Jobs table as well as adding a foreign key.
Following is the query:

```
--22P-9216
ALTER TABLE Jobs ADD Employee_id NUMBER(5);
ALTER TABLE Jobs ADD CONSTRAINT employee_fk FOREIGN KEY (Employee_id) REFERENCES Employees(Employee_id);
```

Following is the result:

```
Table JOBS altered.


Table JOBS altered.
```

Task 13:

Insert 3 rows of data into jobs table.

Ans:

We have inserted 3 rows of data into the Jobs table by using “insert into” and “values”.
Following is the query:

```
--22P-9216
INSERT INTO Jobs (Job_ID, Job_Title, Min_Salary, Max_Salary, Employee_id)
VALUES (101, 'Software Engineer', 50000, 100000, 2001);

INSERT INTO Jobs (Job_ID, Job_Title, Min_Salary, Max_Salary, Employee_id)
VALUES (102, 'HR Manager', 40000, 90000, 2002);

INSERT INTO Jobs (Job_ID, Job_Title, Min_Salary, Max_Salary, Employee_id)
VALUES (103, 'Data Analyst', 45000, 95000, 2003);
```

Following is the result:

```
1 row inserted.


1 row inserted.


1 row inserted.
```

Task 14:

Drop column speciality from Departments.

Ans:

Again “alter” command is used to drop the “Speciality” column.
Following is the query:

```
--22P-9216
ALTER TABLE Departments DROP COLUMN Speciality;
```

Following is the result:

```
Table DEPARTMENTS altered.
```

Task 15:

Truncate the jobs table.

Ans:

To truncate the Jobs table, we use “truncate”.
Following is the command:

```
--22P-9216
TRUNCATE TABLE Jobs;
```

Following is the resut:

```
Table JOBS truncated.
```

Task 16:

Insert 4 new rows into jobs table.

Ans:

Again “insert into” and “values” are used to insert 4 more rows into the Jobs table.
Following is the query:

```
--22P-9216
INSERT INTO Jobs (Job_ID, Job_Title, Min_Salary, Max_Salary, Employee_id)
VALUES (3001, 'Network Administrator', 48000, 92000, 2001);

INSERT INTO Jobs (Job_ID, Job_Title, Min_Salary, Max_Salary, Employee_id)
VALUES (3002, 'Project Manager', 60000, 120000, 2002);

INSERT INTO Jobs (Job_ID, Job_Title, Min_Salary, Max_Salary, Employee_id)
VALUES (3003, 'UI/UX Designer', 47000, 90000, 2003);

INSERT INTO Jobs (Job_ID, Job_Title, Min_Salary, Max_Salary, Employee_id)
VALUES (3004, 'System Analyst', 53000, 97000, 2004);
```

Following is the result:

```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.
```

Task 17:

ALTER table EMPLOYEE and apply the constraint CHECK on Full_Name attribute such that it should always be capitalized.

Ans:

"Check" constraint is added by altering the Employees table.
Following is the query:

```
--22P-9216
ALTER TABLE Employees ADD CONSTRAINT check_upper CHECK (Full_Name=UPPER(Full_Name));
```

Following is the result:

```
Table EMPLOYEES altered.
```

Task 18:

Change table Employee and make sure that Phone_No should be unique, and never empty.

Ans:

To make sure that the phone_no is both unique and empty, we change the Employees table using "alter" command.
Following is the query:

```
--22P-9216
ALTER TABLE Employees MODIFY Phone_No VARCHAR2(20) NOT NULL;

ALTER TABLE Employees ADD CONSTRAINT unique_phone UNIQUE(Phone_No);
```

Following is the result:

```
Table EMPLOYEES altered.


Table EMPLOYEES altered.
```

Task 19:

Write a SQL statement to insert one row into the table Employees.

Ans:

We use “insert into” and “values” to insert another row into the Employees table.
Following is the query:

```
--22P-9216
INSERT INTO Employees (
    Employee_id, Full_Name, Salary, Department_id, Start_Date, End_Date, Married, Phone_No
)
VALUES (
    2006, 'ALI KHAN', 9999.99, 1001, TO_DATE('2023-05-15', 'YYYY-MM-DD'), NULL, 'Y', '0312-9876543'
);
```

Following is the result:

```
1 row inserted.
```

Task 20:

Write a SQL statement to increase the salary of an employee by 200% if the existing salary is less than 1000.

Ans:

We use the “update” and “set” commands to modify the Salary column of the Employees table.
Following is the query:

```
--22P-9216
UPDATE Employees SET Salary=Salary*3 WHERE Salary<1000;
```

Following is the result:

```
0 rows updated.
```

Task 21:

Change column name Phone_No to Phone_Number, and change jobs table to be job_details, make sure to change foreign keys where referenced.

Ans:

To change the name of the “Phone_No” column to “Phone_Number”, we use “alter” command.
Following is the query:

```
--22P-9216
ALTER TABLE Employees RENAME COLUMN Phone_No TO Phone_Number;
```

Following is the result:

```
Table EMPLOYEES altered.
```

Do the same for Jobs table.
Following is the query:

```
--22P-9216
ALTER TABLE Jobs RENAME TO Job_Details;
```

Following is the result:

```
Table JOBS altered.
```

Task 22:

Write a SQL statement to add a primary key for a combination of columns employee_id and job_id in employees table, give the reason why this command is showing error.

Ans:

The error occurs because Employee_ID is already a primary key in the Employees table. A table can have only one primary key, and adding a composite primary key without first removing the existing one causes a conflict.
Following is the query:

```
--22P-9216
ALTER TABLE Employees ADD CONSTRAINT EMPLOYEE_PK PRIMARY KEY (Employee_ID, Job_ID);
```

Task 23:

Delete a row from jobs_details table where starting year is below 1990 (add a record first if not existent).

Ans:

First we insert records into Job_Details table.
Following is the query:

```
--22P-9216
INSERT INTO Job_Details (Job_ID, Employee_ID, Job_Title, Min_Salary, Max_Salary, Employment_Date)
VALUES (5002, 2005, 'Legacy System Engineer', 30000, 60000, TO_DATE('1985-06-15', 'YYYY-MM-DD'));
```

Then we delete those rows.
Following is the query:

```
--22P-9216
DELETE FROM Job_Details WHERE EXTRACT(YEAR FROM Employment_Date) < 1990;
```

Following is the result:

```
1 row deleted.
```

Task 24:

Drop the job_details table.

Ans:

The job_details table is dropped using the “drop” command and we use “cascade constraints” so that the foreign key constraints are also removed.
Following is the query:

```
--22P-9216
DROP TABLE Job_Details CASCADE CONSTRAINTS;
```

Following is the result:

```
Table JOB_DETAILS dropped.
```

Task 25:

Write a SQL statement to add an index named indx_employee_id on employee_id column in the table employees, indx_department_id on department_id column in the table departments.

Ans:

The indexes for primary key or unique constraints are already created. Since Employee_id and Department_id are primary keys, Oracle automatically created these indexes.

Following is the query:

```sql
--22P-9216
CREATE INDEX indx_employee_id ON Employees(employee_id);
CREATE INDEX indx_department_id ON Departments(department_id);
```

Following is the result:

```
Error starting at line : 2 in command -
CREATE INDEX indx_employee_id ON Employees(employee_id)
Error report -
ORA-01408: such column list already indexed
01408. 00000 -  "such column list already indexed"
*Cause:
*Action:

Error starting at line : 3 in command -
CREATE INDEX indx_department_id ON Departments(department_id)
Error report -
ORA-01408: such column list already indexed
01408. 00000 -  "such column list already indexed"
*Cause:
*Action:
```

Task 26:

Create a table named Suppliers with the following fields:Supplier_ID (Primary Key) Supplier_Name (NOT NULL, Unique),Contact_Name, Phone_Number (NOT NULL), Email (Unique).

Ans:

Suppliers table is created using the following query:

```sql
--22P-9216
CREATE TABLE Suppliers(
    Supplier_ID NUMBER(10) PRIMARY KEY,
    Supplier_Name VARCHAR2(50) NOT NULL UNIQUE,
    Contact_Name VARCHAR2(50),
    Phone_Number VARCHAR2(15) NOT NULL,
    Email VARCHAR2(50) UNIQUE
);
```

Following is the result:

```
Table SUPPLIERS created.
```

Task 27:

Create a table named `Products` with the following fields: `Product_ID` (Primary Key), `Product_Name` (NOT NULL, Unique), `Supplier_ID` (Foreign Key referencing the `Suppliers` table), and `Category_ID`.)

Ans:

Products table is created using the following query:

```
--22P-9216
CREATE TABLE Products(
    Product_ID NUMBER(10) PRIMARY KEY,
    Product_Name VARCHAR2(40) NOT NULL UNIQUE,
    Supplier_ID NUMBER(10),
    Category_ID NUMBER(10),
    CONSTRAINT supplier_fk FOREIGN KEY (Supplier_ID) REFERENCES Suppliers(Supplier_ID)
);
```

Following is the result:

```
Table PRODUCTS created.
```

Task 28:

Add 5 Records in Both Tables.

Ans:

Now we add 5 records in the Suppliers table using “insert into”.
Following is the query:

```
--22P-9216
INSERT INTO Suppliers (Supplier_ID, Supplier_Name, Contact_Name, Phone_Number, Email)
VALUES (101, 'Alpha Distributors', 'John Doe', '0300-1234567', 'alpha@suppliers.com');

INSERT INTO Suppliers (Supplier_ID, Supplier_Name, Contact_Name, Phone_Number, Email)
VALUES (102, 'Beta Traders', 'Jane Smith', '0311-9876543', 'beta@suppliers.com');

INSERT INTO Suppliers (Supplier_ID, Supplier_Name, Contact_Name, Phone_Number, Email)
VALUES (103, 'Gamma Supplies', 'Michael Lee', '0322-5678912', 'gamma@suppliers.com');

INSERT INTO Suppliers (Supplier_ID, Supplier_Name, Contact_Name, Phone_Number, Email)
VALUES (104, 'Delta Imports', 'Emily Davis', '0333-1122334', 'delta@suppliers.com');

INSERT INTO Suppliers (Supplier_ID, Supplier_Name, Contact_Name, Phone_Number, Email)
VALUES (105, 'Epsilon Wholesalers', 'Robert Brown', '0344-4455667', 'epsilon@suppliers.com');
```

Following is the result:

```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.
```

Do the same for Products table.

Following is the query:

```
--22P-9216
INSERT INTO Products (Product_ID, Product_Name, Supplier_ID, Category_ID)
VALUES (201, 'Laptop', 101, 1);

INSERT INTO Products (Product_ID, Product_Name, Supplier_ID, Category_ID)
VALUES (202, 'Smartphone', 102, 2);

INSERT INTO Products (Product_ID, Product_Name, Supplier_ID, Category_ID)
VALUES (203, 'Headphones', 103, 3);

INSERT INTO Products (Product_ID, Product_Name, Supplier_ID, Category_ID)
VALUES (204, 'Monitor', 104, 1);

INSERT INTO Products (Product_ID, Product_Name, Supplier_ID, Category_ID)
VALUES (205, 'Keyboard', 105, 3);
```

```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.
```