



SISTER NIVEDITA UNIVERSITY



DATABASE MANAGEMENT SYSTEM

SUBMITTED BY: PIYUSH CHANDRA CHANDRA

DEPT- BTECH (CSE), ROLL- 1027

SUBMITTED TO: SWARUP KUMAR GHOSH &

DEBANJAN DAS

ASSIGNMENT 4

04/11/2020 – 08/11/2020

1. Create table **EMPLOYEE** with the following details.

| FIELD NAME | TYPE |
|---------------|--------------|
| EMPLOYEE_ID | NUMBER(6) |
| LAST_NAME | VARCHAR2(25) |
| JOB_ID | VARCHAR2(10) |
| SALARY | NUMBER(8,2) |
| COMM_PCT | NUMBER (4,2) |
| MGR_ID | NUMBER (6) |
| DEPARTMENT_ID | NUMBER (4) |

➤ CREATE TABLE EMPLOYEE

(

EMPLOYEE_ID NUMBER(6),

LAST_NAME VARCHAR2(25),

JOB_ID VARCHAR2(10),

SALARY NUMBER(8,2),

COMM_PCT NUMBER (4,2),

MGR_ID NUMBER (6),

DEPARTMENT_ID NUMBER (4)

);

ALTER TABLE EMPLOYEE ADD CONSTRAINTS E1 PRIMARY KEY(EMPLOYEE_ID);

desc EMPLOYEE;

The screenshot shows the SQL Developer interface with the following content:

```

CREATE TABLE EMPLOYEE
(
  EMPLOYEE_ID NUMBER(6),
  LAST_NAME VARCHAR2(25),
  JOB_ID VARCHAR2(10),
  SALARY NUMBER(8,2),
  COMM_PCT NUMBER (4,2),
  MGR_ID NUMBER (6),
  DEPARTMENT_ID NUMBER (4)
);
ALTER TABLE EMPLOYEE ADD CONSTRAINTS E1 PRIMARY KEY(EMPLOYEE_ID);
desc EMPLOYEE;
INSERT INTO EMPLOYEE VALUES(198, 'Connell', 'SH_CLERK', 2600, 2.50, 124, 50);
INSERT INTO EMPLOYEE VALUES(199, 'Grant', 'SH_CLERK', 2600, 2.20, 124, 50);
INSERT INTO EMPLOYEE VALUES(200, 'Halen', 'AD_ASST', 4400, 1.30, 101, 10);
INSERT INTO EMPLOYEE VALUES(201, 'Hartstein', 'IT_PROG', 6000, null, 100, 20);
INSERT INTO EMPLOYEE VALUES(202, 'Fav...', 'AC MGR', 6580, null, 210, 20);
  
```

Below the SQL commands, the 'Describe' tab is selected, showing the table structure:

| Table | Column | Data Type | Length | Precision | Scale | Primary Key | Nullable | Default | Comment |
|----------|---------------|-----------|--------|-----------|-------|-------------|----------|---------|---------|
| EMPLOYEE | EMPLOYEE_ID | NUMBER | - | 6 | 0 | 1 | - | - | - |
| | LAST_NAME | VARCHAR2 | 25 | - | - | - | ✓ | - | - |
| | JOB_ID | VARCHAR2 | 10 | - | - | - | ✓ | - | - |
| | SALARY | NUMBER | - | 8 | 2 | - | ✓ | - | - |
| | COMM_PCT | NUMBER | - | 4 | 2 | - | ✓ | - | - |
| | MGR_ID | NUMBER | - | 6 | 0 | - | ✓ | - | - |
| | DEPARTMENT_ID | NUMBER | - | 4 | 0 | - | ✓ | - | - |

2. Insert the following data into EMPLOYEE table.

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |
| 200 | Whalen | AD_ASST | 4400 | 1.3 | 101 | 10 |
| 201 | Hartstein | IT_PROG | 6000 | null | 100 | 20 |
| 202 | Fay | AC_MGR | 6500 | null | 210 | 20 |
| 203 | Mavris | AD_VP | 7500 | null | 101 | 40 |
| 204 | Baer | AD_PRES | 3500 | 1.5 | 101 | 90 |
| 205 | Higgins | AC_MGR | 2300 | null | 101 | 60 |
| 206 | Gitz | IT_PROG | 5000 | null | 103 | 60 |
| 100 | King | AD_ASST | 8956 | 0.3 | 108 | 100 |
| 101 | Kocher | SH_CLERK | 3400 | 1.3 | 118 | 30 |

➤ INSERT INTO EMPLOYEE VALUES(198 , 'Connell', 'SH_CLERK', 2600 , 2.50 , 124 , 50)

INSERT INTO EMPLOYEE VALUES(199 , 'Grant', 'SH_CLERK' ,2600, 2.20 , 124 , 50)

INSERT INTO EMPLOYEE VALUES(200 , 'Whalen', 'AD_ASST' ,4400 , 1.30 , 101 , 10)

INSERT INTO EMPLOYEE VALUES(201 , 'Hartstein','IT_PROG', 6000 , null , 100 , 20)

INSERT INTO EMPLOYEE VALUES(202 , 'Fay' , 'AC_MGR' ,6500 , null , 210 , 20)

INSERT INTO EMPLOYEE VALUES(203 , 'Mavris' , 'AD_VP', 7500, null , 101 , 40)

INSERT INTO EMPLOYEE VALUES(204 , 'Baer', 'AD_PRES' ,3500 , 1.50 , 101 , 90)

INSERT INTO EMPLOYEE VALUES(205 , 'Higgins', 'AC_MGR' ,2300 , null , 101 , 60)

INSERT INTO EMPLOYEE VALUES(206 , 'Gitz', 'IT_PROG' ,5000 ,null , 103 , 60)

INSERT INTO EMPLOYEE VALUES(100 , 'King', 'AD_ASST' ,8956 , 0.30 , 108 , 100)

INSERT INTO EMPLOYEE VALUES(101 , 'Kocher', 'SH_CLERK' ,3400 , 1.30, 118 , 30)

SELECT * FROM EMPLOYEE;

Home > SQL Workshop > SQL Commands

Autocommit Rows 100 Save Run

```

MGR_ID NUMBER (6),
DEPARTMENT_ID NUMBER (4)
);
ALTER TABLE EMPLOYEE ADD CONSTRAINTS E1 PRIMARY KEY(EMPLOYEE_ID);
desc EMPLOYEE;
INSERT INTO EMPLOYEE VALUES(198, 'Connell', 'SH_CLERK', 2600, 2.50, 124, 50 )
INSERT INTO EMPLOYEE VALUES(199, 'Grant', 'SH_CLERK', 2600, 2.20, 124, 50 )
INSERT INTO EMPLOYEE VALUES(200, 'Whalen', 'AD_ASST', 4400, 1.30, 101, 10 )
INSERT INTO EMPLOYEE VALUES(201, 'Hartstein', 'IT_PROG', 6000, null, 100, 20 )
INSERT INTO EMPLOYEE VALUES(202, 'Fay', 'AC_MGR', 6500, null, 210, 20 )
INSERT INTO EMPLOYEE VALUES(203, 'Mavris', 'AD_VP', 7500, null, 101, 40 )
INSERT INTO EMPLOYEE VALUES(204, 'Baer', 'AD_PRES', 3500, 1.50, 101, 90 )
INSERT INTO EMPLOYEE VALUES(205, 'Higgins', 'AC_MGR', 2300, null, 101, 60 )
INSERT INTO EMPLOYEE VALUES(206, 'Gitz', 'IT_PROG', 5000, null, 103, 60 )
INSERT INTO EMPLOYEE VALUES(100, 'King', 'AD_ASST', 8956, 0.30, 108, 100 )
INSERT INTO EMPLOYEE VALUES(101, 'Kocher', 'SH_CLERK', 3400, 1.30, 118, 30 )
SELECT * FROM EMPLOYEE;

```

Results Explain Describe Saved SQL History

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |
| 200 | Whalen | AD_ASST | 4400 | 1.3 | 101 | 10 |
| 201 | Hartstein | IT_PROG | 6000 | - | 100 | 20 |
| 202 | Fay | AC_MGR | 6500 | - | 210 | 20 |
| 203 | Mavris | AD_VP | 7500 | - | 101 | 40 |
| 100 | King | AD_ASST | 8956 | .3 | 108 | 100 |
| 101 | Kocher | SH_CLERK | 3400 | 1.3 | 118 | 30 |
| 204 | Baer | AD_PRES | 3500 | 1.5 | 101 | 90 |
| 205 | Higgins | AC_MGR | 2300 | - | 101 | 60 |
| 206 | Gitz | IT_PROG | 5000 | - | 103 | 60 |

11 rows returned in 0.01 seconds Download

3. Display last_name, job_id, employee_id for each employee with employee_id appearing first.

- SELECT EMPLOYEE_ID, LAST_NAME, JOB_ID
FROM EMPLOYEE

Home > SQL Workshop > SQL Commands

Autocommit Rows 100 Save Run

```

ALTER TABLE EMPLOYEE ADD CONSTRAINTS E1 PRIMARY KEY(EMPLOYEE_ID);
desc EMPLOYEE;
INSERT INTO EMPLOYEE VALUES(198, 'Connell', 'SH_CLERK', 2600, 2.50, 124, 50 )
INSERT INTO EMPLOYEE VALUES(199, 'Grant', 'SH_CLERK', 2600, 2.20, 124, 50 )
INSERT INTO EMPLOYEE VALUES(200, 'Whalen', 'AD_ASST', 4400, 1.30, 101, 10 )
INSERT INTO EMPLOYEE VALUES(201, 'Hartstein', 'IT_PROG', 6000, null, 100, 20 )
INSERT INTO EMPLOYEE VALUES(202, 'Fay', 'AC_MGR', 6500, null, 210, 20 )
INSERT INTO EMPLOYEE VALUES(203, 'Mavris', 'AD_VP', 7500, null, 101, 40 )
INSERT INTO EMPLOYEE VALUES(204, 'Baer', 'AD_PRES', 3500, 1.50, 101, 90 )
INSERT INTO EMPLOYEE VALUES(205, 'Higgins', 'AC_MGR', 2300, null, 101, 60 )
INSERT INTO EMPLOYEE VALUES(206, 'Gitz', 'IT_PROG', 5000, null, 103, 60 )
INSERT INTO EMPLOYEE VALUES(100, 'King', 'AD_ASST', 8956, 0.30, 108, 100 )
INSERT INTO EMPLOYEE VALUES(101, 'Kocher', 'SH_CLERK', 3400, 1.30, 118, 30 )
SELECT * FROM EMPLOYEE;
SELECT EMPLOYEE_ID, LAST_NAME, JOB_ID
FROM EMPLOYEE;

```

Results Explain Describe Saved SQL History

| EMPLOYEE_ID | LAST_NAME | JOB_ID |
|-------------|-----------|----------|
| 198 | Connell | SH_CLERK |
| 199 | Grant | SH_CLERK |
| 200 | Whalen | AD_ASST |
| 201 | Hartstein | IT_PROG |
| 202 | Fay | AC_MGR |
| 203 | Mavris | AD_VP |
| 100 | King | AD_ASST |
| 101 | Kocher | SH_CLERK |
| 204 | Baer | AD_PRES |
| 205 | Higgins | AC_MGR |
| 206 | Gitz | IT_PROG |

11 rows returned in 0.00 seconds Download

4. Display the details of all employees of department 60.

```
➤ SELECT *  
FROM EMPLOYEE  
WHERE DEPARTMENT_ID = 60
```

The screenshot shows the SQL Workshop interface with the following SQL commands:

```
INSERT INTO EMPLOYEE VALUES(202, 'Fay', 'AC_MGR', 6500, null, 210, 20 )  
INSERT INTO EMPLOYEE VALUES(203, 'Mavris', 'AD_VP', 7500, null, 101, 40 )  
INSERT INTO EMPLOYEE VALUES(204, 'Beers', 'AD_PRES', 3500, 1.50, 101, 90 )  
INSERT INTO EMPLOYEE VALUES(205, 'Higgins', 'AC_MGR', 2300, null, 101, 60 )  
INSERT INTO EMPLOYEE VALUES(206, 'Glitz', 'IT_PROG', 5000, null, 103, 60 )  
INSERT INTO EMPLOYEE VALUES(100, 'King', 'AD_ASST', 8956, 0.30, 108, 100 )  
INSERT INTO EMPLOYEE VALUES(101, 'Kochhar', 'SH_CLERK', 3400, 1.30, 118, 30 )  
SELECT * FROM EMPLOYEE;  
  
SELECT EMPLOYEE_ID, LAST_NAME, JOB_ID  
FROM EMPLOYEE  
SELECT *  
FROM EMPLOYEE  
WHERE DEPARTMENT_ID = 60  
SELECT *  
FROM EMPLOYEE
```

The Results tab shows the following data:

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|---------|--------|----------|--------|---------------|
| 205 | Higgins | AC_MGR | 2300 | - | 101 | 60 |
| 206 | Glitz | IT_PROG | 5000 | - | 103 | 60 |

2 rows returned in 0.01 seconds

5. Display the employee details of the employee whose last_name is King.

```
➤ SELECT *  
FROM EMPLOYEE  
WHERE LAST_NAME = 'King';
```

The screenshot shows the SQL Workshop interface with the following SQL commands:

```
INSERT INTO EMPLOYEE VALUES(202, 'Fay', 'AC_MGR', 6500, null, 210, 20 )  
INSERT INTO EMPLOYEE VALUES(203, 'Mavris', 'AD_VP', 7500, null, 101, 40 )  
INSERT INTO EMPLOYEE VALUES(204, 'Beers', 'AD_PRES', 3500, 1.50, 101, 90 )  
INSERT INTO EMPLOYEE VALUES(205, 'Higgins', 'AC_MGR', 2300, null, 101, 60 )  
INSERT INTO EMPLOYEE VALUES(206, 'Glitz', 'IT_PROG', 5000, null, 103, 60 )  
INSERT INTO EMPLOYEE VALUES(100, 'King', 'AD_ASST', 8956, 0.30, 108, 100 )  
INSERT INTO EMPLOYEE VALUES(101, 'Kochhar', 'SH_CLERK', 3400, 1.30, 118, 30 )  
SELECT * FROM EMPLOYEE;  
  
SELECT EMPLOYEE_ID, LAST_NAME, JOB_ID  
FROM EMPLOYEE  
SELECT *  
FROM EMPLOYEE  
WHERE DEPARTMENT_ID = 60  
SELECT *  
FROM EMPLOYEE  
WHERE LAST_NAME = 'King'
```

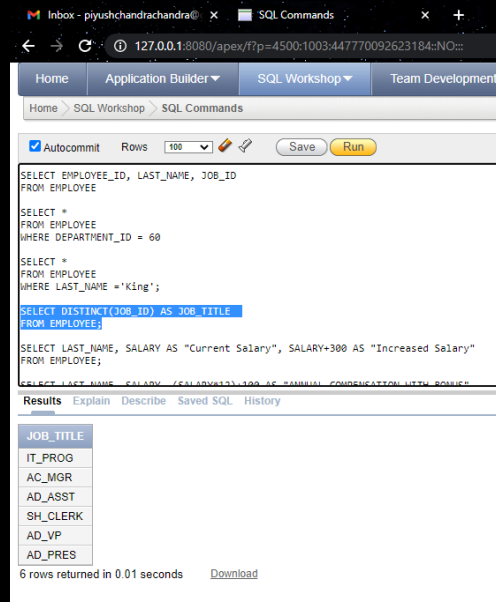
The Results tab shows the following data:

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|---------|--------|----------|--------|---------------|
| 100 | King | AD_ASST | 8956 | .3 | 108 | 100 |

1 rows returned in 0.00 seconds

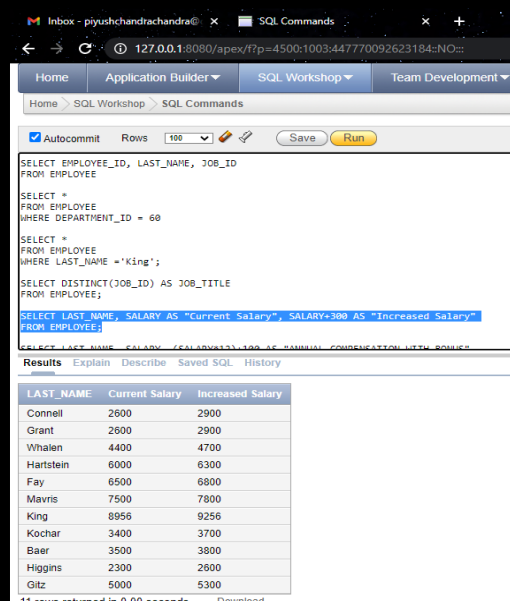
6. Display unique job_id from EMPLOYEE table. Give alias name to the column as JOB_TITLE.

- SELECT DISTINCT(JOB_ID) AS JOB_TITLE
FROM EMPLOYEE;



7. Display last_name, salary and salary increase of Rs300. Give the new column name as 'Increased Salary'.

- SELECT LAST_NAME, SALARY AS "Current Salary", SALARY+300 AS "Increased Salary"
FROM EMPLOYEE;



8. Display last name, salary and annual compensation of all employees, plus a onetime bonus of Rs 100. Give an alias name to the column displaying annual compensation.

- SELECT LAST_NAME, SALARY, (SALARY*12)+100 AS "ANNUAL COMPENSATION WITH BONUS"
FROM EMPLOYEE;

The screenshot shows the SQL Workshop interface with the following SQL query entered:

```
SELECT *  
FROM EMPLOYEE  
WHERE LAST_NAME = 'King';  
  
SELECT DISTINCT(JOB_ID) AS JOB_TITLE  
FROM EMPLOYEE;  
  
SELECT LAST_NAME, SALARY AS "Current Salary", SALARY+300 AS "Increased Salary"  
FROM EMPLOYEE;  
  
SELECT LAST_NAME, SALARY, (SALARY*12)+100 AS "ANNUAL COMPENSATION WITH BONUS"  
FROM EMPLOYEE;  
  
SELECT *  
FROM EMPLOYEE  
WHERE COMM_PCT IS NOT NULL;
```

The results of the query are displayed in a table:

| LAST_NAME | SALARY | ANNUAL COMPENSATION WITH BONUS |
|-----------|--------|--------------------------------|
| Connell | 2600 | 31300 |
| Grant | 2600 | 31300 |
| Whalen | 4400 | 52900 |
| Hartstein | 6000 | 72100 |
| Fay | 6500 | 78100 |
| Mavris | 7500 | 90100 |
| King | 8956 | 107572 |
| Kocher | 3400 | 40900 |
| Baer | 3500 | 42100 |
| Higgins | 2300 | 27700 |
| Giltz | 5000 | 60100 |

11 rows returned in 0.00 seconds

9. Display the details of those employees who get commission.

- SELECT *
FROM EMPLOYEE
WHERE COMM_PCT IS NOT NULL;

The screenshot shows the SQL Workshop interface with the following SQL query entered:

```
SELECT *  
FROM EMPLOYEE  
WHERE LAST_NAME = 'King';  
  
SELECT DISTINCT(JOB_ID) AS JOB_TITLE  
FROM EMPLOYEE;  
  
SELECT LAST_NAME, SALARY AS "Current Salary", SALARY+300 AS "Increased Salary"  
FROM EMPLOYEE;  
  
SELECT LAST_NAME, SALARY, (SALARY*12)+100 AS "ANNUAL COMPENSATION WITH BONUS"  
FROM EMPLOYEE;  
  
SELECT *  
FROM EMPLOYEE  
WHERE COMM_PCT IS NOT NULL;
```

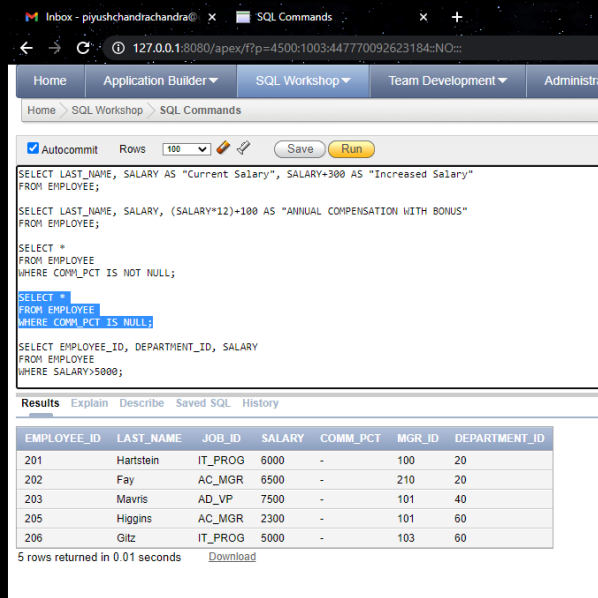
The results of the query are displayed in a table:

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |
| 200 | Whalen | AD_ASST | 4400 | 1.3 | 101 | 10 |
| 100 | King | AD_ASST | 8956 | .3 | 108 | 100 |
| 101 | Kocher | SH_CLERK | 3400 | 1.3 | 118 | 30 |
| 204 | Baer | AD_PRES | 3500 | 1.5 | 101 | 90 |

6 rows returned in 0.01 seconds

10. Display the details of those employees who do not get commission.

➤ SELECT *
FROM EMPLOYEE
WHERE COMM_PCT IS NULL;



The screenshot shows the SQL Workshop interface with the following SQL query entered:

```
SELECT LAST_NAME, SALARY AS "Current Salary", SALARY+300 AS "Increased Salary"  
FROM EMPLOYEE;  
  
SELECT LAST_NAME, SALARY, (SALARY*12)+100 AS "ANNUAL COMPENSATION WITH BONUS"  
FROM EMPLOYEE;  
  
SELECT *  
FROM EMPLOYEE  
WHERE COMM_PCT IS NOT NULL;  
  
SELECT *  
FROM EMPLOYEE  
WHERE COMM_PCT IS NULL;  
  
SELECT EMPLOYEE_ID, DEPARTMENT_ID, SALARY  
FROM EMPLOYEE  
WHERE SALARY>5000;
```

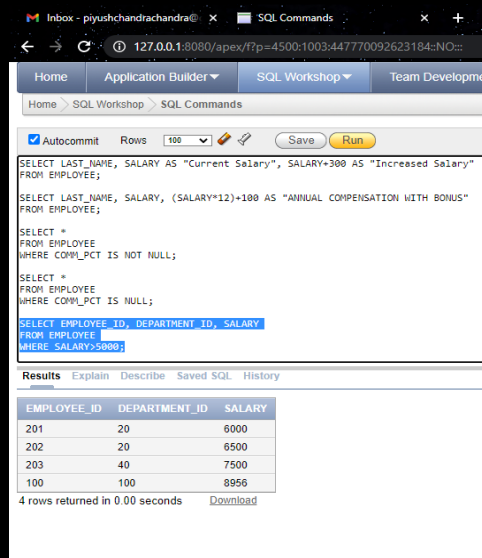
The results of the query are displayed in a table with 5 rows:

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|---------|--------|----------|--------|---------------|
| 201 | Hartstein | IT_PROG | 6000 | - | 100 | 20 |
| 202 | Fay | AC_MGR | 6500 | - | 210 | 20 |
| 203 | Mavris | AD_VP | 7500 | - | 101 | 40 |
| 205 | Higgins | AC_MGR | 2300 | - | 101 | 60 |
| 206 | Gitz | IT_PROG | 5000 | - | 103 | 60 |

5 rows returned in 0.01 seconds

11. Display the Employee_id, Department_id and Salary all employees whose salary is greater than 5000.

➤ SELECT EMPLOYEE_ID, DEPARTMENT_ID, SALARY
FROM EMPLOYEE
WHERE SALARY>5000;



The screenshot shows the SQL Workshop interface with the following SQL query entered:

```
SELECT LAST_NAME, SALARY AS "Current Salary", SALARY+300 AS "Increased Salary"  
FROM EMPLOYEE;  
  
SELECT LAST_NAME, SALARY, (SALARY*12)+100 AS "ANNUAL COMPENSATION WITH BONUS"  
FROM EMPLOYEE;  
  
SELECT *  
FROM EMPLOYEE  
WHERE COMM_PCT IS NOT NULL;  
  
SELECT *  
FROM EMPLOYEE  
WHERE COMM_PCT IS NULL;  
  
SELECT EMPLOYEE_ID, DEPARTMENT_ID, SALARY  
FROM EMPLOYEE  
WHERE SALARY>5000;
```

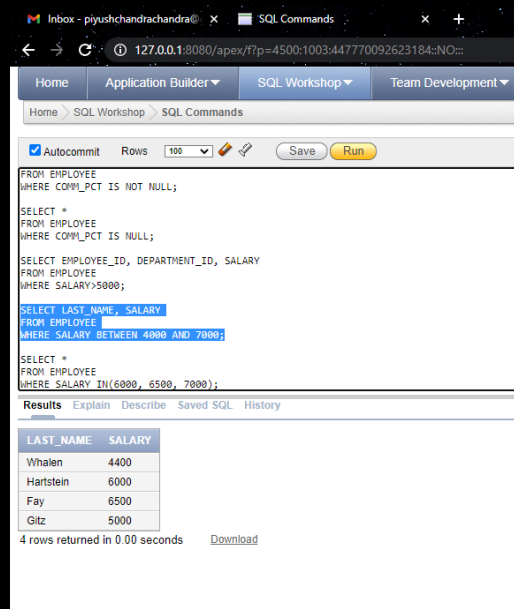
The results of the query are displayed in a table with 4 rows:

| EMPLOYEE_ID | DEPARTMENT_ID | SALARY |
|-------------|---------------|--------|
| 201 | 20 | 6000 |
| 202 | 20 | 6500 |
| 203 | 40 | 7500 |
| 100 | 100 | 8956 |

4 rows returned in 0.00 seconds

12.Display the Last Name and Salary of all employees whose salary is between 4000 and 7000.

- SELECT LAST_NAME, SALARY
FROM EMPLOYEE
WHERE SALARY BETWEEN 4000 AND 7000;



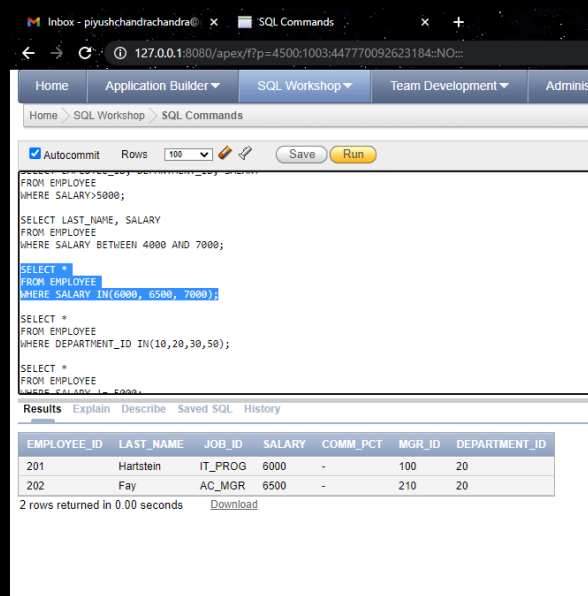
The screenshot shows the SQL Workshop interface with a query editor and a results table. The query is: `SELECT LAST_NAME, SALARY FROM EMPLOYEE WHERE SALARY BETWEEN 4000 AND 7000;`. The results table has two columns: LAST_NAME and SALARY. It contains four rows of data.

| LAST_NAME | SALARY |
|-----------|--------|
| Whalen | 4400 |
| Hartstein | 6000 |
| Fay | 6500 |
| Gitz | 5000 |

4 rows returned in 0.00 seconds

13.Display the details of all employees whose salary is either 6000 or 6500 or 7000.

- SELECT *
FROM EMPLOYEE
WHERE SALARY IN(6000, 6500, 7000);



The screenshot shows the SQL Workshop interface with a query editor and a results table. The query is: `SELECT * FROM EMPLOYEE WHERE SALARY IN(6000, 6500, 7000);`. The results table has seven columns: EMPLOYEE_ID, LAST_NAME, JOB_ID, SALARY, COMM_PCT, MGR_ID, and DEPARTMENT_ID. It contains two rows of data.

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|---------|--------|----------|--------|---------------|
| 201 | Hartstein | IT_PROG | 6000 | - | 100 | 20 |
| 202 | Fay | AC_MGR | 6500 | - | 210 | 20 |

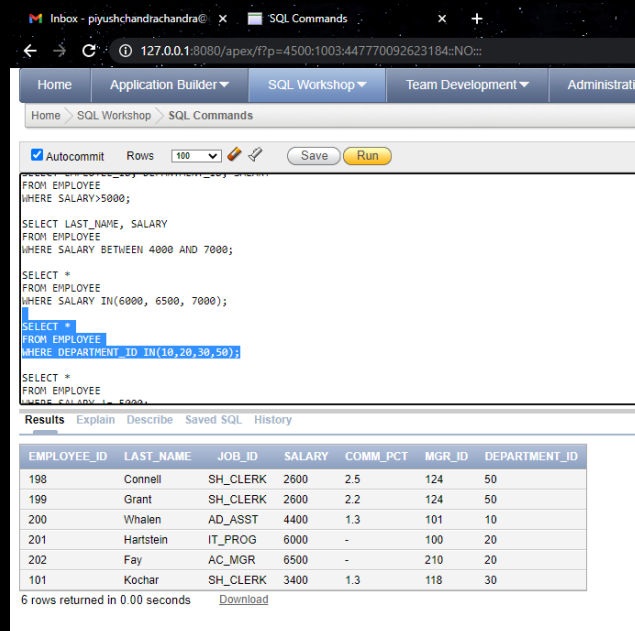
2 rows returned in 0.00 seconds

14. Display the details of all those employees who work either in department 10 or 20 or 30 or 50.

➤ SELECT *

FROM EMPLOYEE

WHERE DEPARTMENT_ID IN(10,20,30,50);



```
FROM EMPLOYEE
WHERE SALARY > 5000;

SELECT LAST_NAME, SALARY
FROM EMPLOYEE
WHERE SALARY BETWEEN 4000 AND 7000;

SELECT *
FROM EMPLOYEE
WHERE SALARY IN(6000, 6500, 7000);

SELECT *
FROM EMPLOYEE
WHERE DEPARTMENT_ID IN(10,20,30,50);

SELECT *
FROM EMPLOYEE
WHERE SALARY < 5000;
```

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |
| 200 | Whalen | AD_ASST | 4400 | 1.3 | 101 | 10 |
| 201 | Hartstein | IT_PROG | 6000 | - | 100 | 20 |
| 202 | Fay | AC_MGR | 6500 | - | 210 | 20 |
| 101 | Kocher | SH_CLERK | 3400 | 1.3 | 118 | 30 |

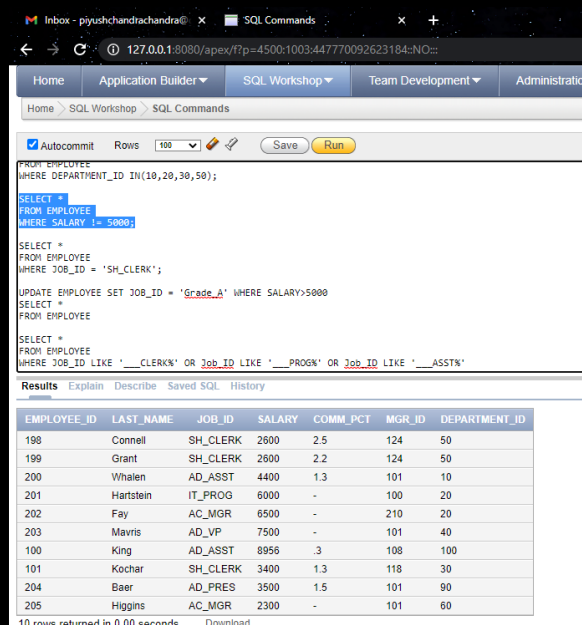
6 rows returned in 0.00 seconds [Download](#)

15. Display the details of all employees whose salary is not equal to 5000.

➤ SELECT *

FROM EMPLOYEE

WHERE SALARY != 5000;



```
FROM EMPLOYEE
WHERE DEPARTMENT_ID IN(10,20,30,50);

SELECT *
FROM EMPLOYEE
WHERE SALARY != 5000;

SELECT *
FROM EMPLOYEE
WHERE JOB_ID = 'SH_CLERK';

UPDATE EMPLOYEE SET JOB_ID = 'Grade_A' WHERE SALARY > 5000

SELECT *
FROM EMPLOYEE

SELECT *
FROM EMPLOYEE
WHERE JOB_ID LIKE '___CLERKS' OR JOB_ID LIKE '___PROG%' OR JOB_ID LIKE '___ASSTS'
```

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |
| 200 | Whalen | AD_ASST | 4400 | 1.3 | 101 | 10 |
| 201 | Hartstein | IT_PROG | 6000 | - | 100 | 20 |
| 202 | Fay | AC_MGR | 6500 | - | 210 | 20 |
| 203 | Mavris | AD_VP | 7500 | - | 101 | 40 |
| 100 | King | AD_ASST | 8956 | .3 | 108 | 100 |
| 101 | Kocher | SH_CLERK | 3400 | 1.3 | 118 | 30 |
| 204 | Baer | AD_PRES | 3500 | 1.5 | 101 | 90 |
| 205 | Higgins | AC_MGR | 2300 | - | 101 | 60 |

10 rows returned in 0.00 seconds [Download](#)

16. Display the details of all the CLERKS working in the organization.

```
➤ SELECT *  
FROM EMPLOYEE  
WHERE JOB_ID = 'SH_CLERK';
```

The screenshot shows the SQL Workshop interface with the following SQL commands entered:

```
FROM EMPLOYEE  
WHERE DEPARTMENT_ID IN(10,20,30,50);  
  
SELECT *  
FROM EMPLOYEE  
WHERE SALARY != 5000;  
  
SELECT *  
FROM EMPLOYEE  
WHERE JOB_ID = 'SH_CLERK';  
  
UPDATE EMPLOYEE SET JOB_ID = 'Grade_A' WHERE SALARY>5000  
SELECT *  
FROM EMPLOYEE  
WHERE JOB_ID LIKE '___CLERK%' OR JOB_ID LIKE '___PROG%' OR JOB_ID LIKE '___ASST%'
```

The results table shows 3 rows returned:

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |
| 101 | Kocher | SH_CLERK | 3400 | 1.3 | 118 | 30 |

17. Update the job_id's of the employees who earn more than 5000 to Grade_A. Display the table EMPLOYEE after updating.

```
➤ UPDATE EMPLOYEE SET JOB_ID = 'Grade_A' WHERE SALARY>5000  
  
SELECT *  
FROM EMPLOYEE  
  
SELECT * FROM EMPLOYEE;
```

The screenshot shows the SQL Workshop interface with the following SQL commands entered:

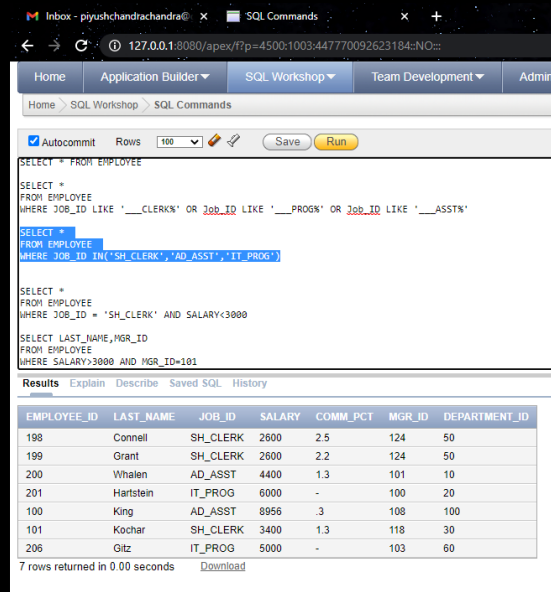
```
FROM EMPLOYEE  
WHERE DEPARTMENT_ID IN(10,20,30,50);  
  
SELECT *  
FROM EMPLOYEE  
WHERE SALARY != 5000;  
  
SELECT *  
FROM EMPLOYEE  
WHERE JOB_ID = 'SH_CLERK';  
  
UPDATE EMPLOYEE SET JOB_ID = 'Grade_A' WHERE SALARY>5000  
SELECT *  
FROM EMPLOYEE  
SELECT * FROM EMPLOYEE;  
  
SELECT *  
FROM EMPLOYEE
```

The results table shows 11 rows returned:

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |
| 200 | Whalen | AD_ASST | 4400 | 1.3 | 101 | 10 |
| 201 | Hartstein | IT_PROG | 6000 | - | 100 | 20 |
| 202 | Fay | AC_MGR | 6500 | - | 210 | 20 |
| 203 | Mavris | AD_VP | 7500 | - | 101 | 40 |
| 100 | King | AD_ASST | 8956 | .3 | 108 | 100 |
| 101 | Kocher | SH_CLERK | 3400 | 1.3 | 118 | 30 |
| 204 | Baer | AD_PRES | 3500 | 1.5 | 101 | 90 |
| 205 | Higgins | AC_MGR | 2300 | - | 101 | 60 |
| 206 | Giltz | IT_PROG | 5000 | - | 103 | 60 |

18. Display the details of all those employees who are either **CLERK** or **PROGRAMMER** or **ASSISTANT**.

➤ **SELECT ***
FROM EMPLOYEE
WHERE JOB_ID IN('SH_CLERK','AD_ASST','IT_PROG')



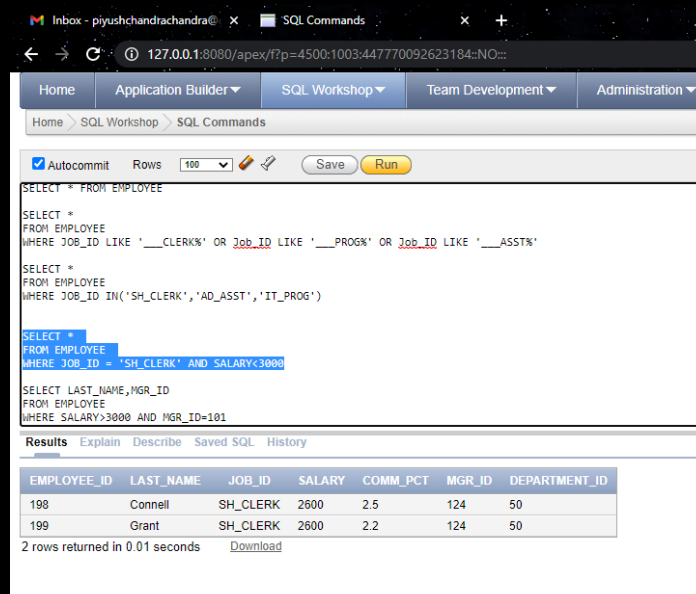
The screenshot shows the SQL Workshop interface with the query: `SELECT * FROM EMPLOYEE WHERE JOB_ID IN('SH_CLERK','AD_ASST','IT_PROG')`. The results table displays 7 rows of employee data.

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |
| 200 | Whalen | AD_ASST | 4400 | 1.3 | 101 | 10 |
| 201 | Hartstein | IT_PROG | 6000 | - | 100 | 20 |
| 100 | King | AD_ASST | 8956 | .3 | 108 | 100 |
| 101 | Kocher | SH_CLERK | 3400 | 1.3 | 118 | 30 |
| 205 | Gitz | IT_PROG | 5000 | - | 103 | 60 |

7 rows returned in 0.00 seconds

19. Display those employees from the **EMPLOYEE** table whose designation is **CLERK** and salary is less than 3000.

➤ **SELECT ***
FROM EMPLOYEE
WHERE JOB_ID = 'SH_CLERK' AND SALARY < 3000



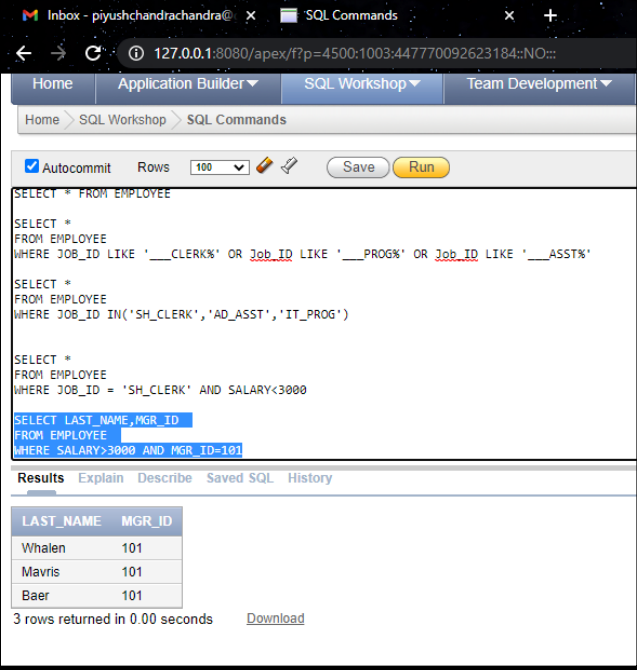
The screenshot shows the SQL Workshop interface with the query: `SELECT * FROM EMPLOYEE WHERE JOB_ID = 'SH_CLERK' AND SALARY < 3000`. The results table displays 2 rows of employee data.

| EMPLOYEE_ID | LAST_NAME | JOB_ID | SALARY | COMM_PCT | MGR_ID | DEPARTMENT_ID |
|-------------|-----------|----------|--------|----------|--------|---------------|
| 198 | Connell | SH_CLERK | 2600 | 2.5 | 124 | 50 |
| 199 | Grant | SH_CLERK | 2600 | 2.2 | 124 | 50 |

2 rows returned in 0.01 seconds

20. Display those employees Last_Name, Mgr_id from the EMPLOYEE table whose salary is above 3000 and work under Manager 101.

➤ SELECT LAST_NAME, MGR_ID
FROM EMPLOYEE
WHERE SALARY > 3000 AND MGR_ID = 101



The screenshot shows the SQL Workshop interface in a web browser. The browser tabs include 'Inbox - piyushchandrachandra@' and 'SQL Commands'. The address bar shows '127.0.0.1:8080/apex/f?p=4500:1003:447770092623184::NO::'. The navigation bar has 'Home', 'Application Builder', 'SQL Workshop', and 'Team Development'. The breadcrumb trail is 'Home > SQL Workshop > SQL Commands'. Below the navigation bar, there are buttons for 'Autocommit' (checked), 'Rows' (set to 100), 'Save', and 'Run'. The SQL editor contains the following query:

```
SELECT * FROM EMPLOYEE  
  
SELECT *  
FROM EMPLOYEE  
WHERE JOB_ID LIKE '___CLERK%' OR JOB_ID LIKE '___PROG%' OR JOB_ID LIKE '___ASST%'  
  
SELECT *  
FROM EMPLOYEE  
WHERE JOB_ID IN ('SH_CLERK', 'AD_ASST', 'IT_PROG')  
  
SELECT *  
FROM EMPLOYEE  
WHERE JOB_ID = 'SH_CLERK' AND SALARY < 3000  
  
SELECT LAST_NAME, MGR_ID  
FROM EMPLOYEE  
WHERE SALARY > 3000 AND MGR_ID = 101
```

Below the SQL editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing a table with two columns: 'LAST_NAME' and 'MGR_ID'. The table contains three rows of data:

| LAST_NAME | MGR_ID |
|-----------|--------|
| Whalen | 101 |
| Mavris | 101 |
| Baer | 101 |

Below the table, it says '3 rows returned in 0.00 seconds' and there is a 'Download' link.