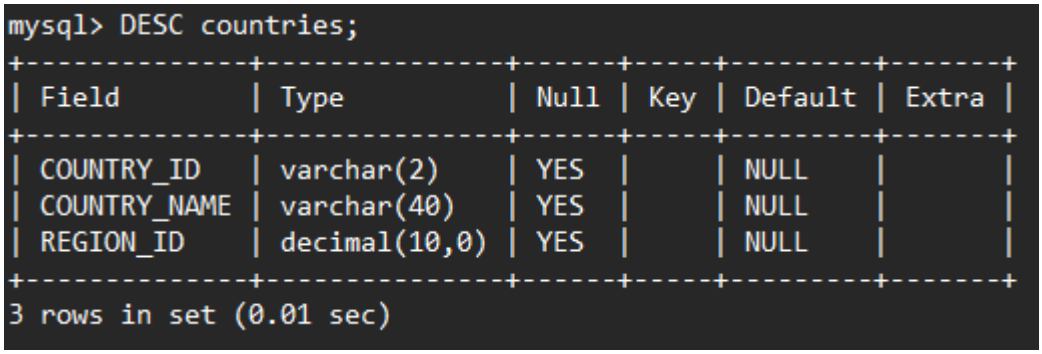
SQL Code Questions

MySQL Create Table Questions

1. Write a SQL statement to create a simple table countries including columns country_id,country_name and region_id.

```
CREATE TABLE countries(
COUNTRY_ID varchar(2),
COUNTRY_NAME varchar(40),
REGION_ID decimal(10,0)
);
DESC countries;
```

Here is the structure of the table:



2. Write a SQL statement to create a simple table countries including columns country_id,country_name and region_id which is already exists.

```
CREATE TABLE IF NOT EXISTS countries (
COUNTRY_ID varchar(2),
COUNTRY_NAME varchar(40),
REGION_ID decimal(10,0)
);
```

Here is the structure of the table:

```
mysql> DESC countries;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
3 rows in set (0.13 sec)
```

3. Write a SQL statement to create the structure of a table dup_countries similar to countries.

```
CREATE TABLE IF NOT EXISTS dup_countries
LIKE countries;
```

Here is the structure of the table:

```
mysql> DESC dup_countries;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
3 rows in set (0.03 sec)
```

4. Write a SQL statement to create a duplicate copy of countries table including structure and data by name dup_countries.

```
CREATE TABLE IF NOT EXISTS dup_countries
AS SELECT * FROM  countries;
```

Here is the structure of the table:

```
mysql> DESC dup_countries;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
3 rows in set (0.11 sec)
```

5. Write a SQL statement to create a table countries set a constraint NULL.

```
CREATE TABLE IF NOT EXISTS countries (
COUNTRY_ID varchar(2) NOT NULL,
COUNTRY_NAME varchar(40) NOT NULL,
REGION_ID decimal(10,0) NOT NULL
);
```

Here is the structure of the table:

```
mysql> desc countries;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | NO   |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | NO   |     | NULL    |       |
| REGION_ID    | decimal(10,0) | NO   |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

6. Write a SQL statement to create a table named jobs including columns job_id, job_title, min_salary, max_salary and check whether the max_salary amount exceeding the upper limit 25000.

```
CREATE TABLE IF NOT EXISTS jobs (
JOB_ID varchar(10) NOT NULL ,
JOB_TITLE varchar(35) NOT NULL,
MIN_SALARY decimal(6,0),
MAX_SALARY decimal(6,0)
CHECK(MAX_SALARY<=25000)
);
```

Here is the structure of the table:

```
mysql> DESC jobs;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| JOB_ID     | varchar(10)  | NO   |     | NULL    |       |
| JOB_TITLE  | varchar(35)  | NO   |     | NULL    |       |
| MIN_SALARY | decimal(6,0) | YES  |     | NULL    |       |
| MAX_SALARY | decimal(6,0) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
4 rows in set (0.16 sec)
```

7. Write a SQL statement to create a table named countries including columns country_id, country_name and region_id and make sure that no countries except Italy, India and China will be entered in the table.

```
CREATE TABLE IF NOT EXISTS countries (
COUNTRY_ID varchar(2),
COUNTRY_NAME varchar(40)
CHECK(COUNTRY_NAME IN('Italy','India','China')) ,
REGION_ID decimal(10,0)
);
```

```
mysql> DESC countries;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

8. Write a SQL statement to create a table named job_histry including columns employee_id, start_date, end_date, job_id and department_id and make sure that the value against column end_date will be entered at the time of insertion to the format like '--/--/----'.

```
CREATE TABLE IF NOT EXISTS job_history (
EMPLOYEE_ID decimal(6,0) NOT NULL,
START_DATE date NOT NULL,
END_DATE date NOT NULL
CHECK (END_DATE LIKE '--/--/----'),
JOB_ID varchar(10) NOT NULL,
DEPARTMENT_ID decimal(4,0) NOT NULL
);
```

Here is the structure of the table:

```
mysql> DESC job_history;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| EMPLOYEE_ID   | decimal(6,0) | NO   |     | NULL    |       |
| START_DATE    | date         | NO   |     | NULL    |       |
| END_DATE      | date         | NO   |     | NULL    |       |
| JOB_ID        | varchar(10)  | NO   |     | NULL    |       |
| DEPARTMENT_ID | decimal(4,0) | NO   |     | NULL    |       |
+---------------+--------------+------+-----+---------+-------+
5 rows in set (0.04 sec)
```

9. Write a SQL statement to create a table named countries including columns country_id,country_name and region_id and make sure that no duplicate data against column country_id will be allowed at the time of insertion.

```
CREATE TABLE IF NOT EXISTS countries (
COUNTRY_ID varchar(2) NOT NULL,
COUNTRY_NAME varchar(40) NOT NULL,
REGION_ID decimal(10,0) NOT NULL,
UNIQUE(COUNTRY_ID)
);
```

Here is the structure of the table:

```
mysql> DESC countries;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

10. Write a SQL statement to create a table named jobs including columns job_id, job_title, min_salary and max_salary, and make sure that, the default value for job_title is blank and min_salary is 8000 and max_salary is NULL will be entered automatically at the time of insertion if no value assigned for the specified columns.

```
CREATE TABLE IF NOT EXISTS jobs (
JOB_ID varchar(10) NOT NULL UNIQUE,
JOB_TITLE varchar(35) NOT NULL DEFAULT ' ',
MIN_SALARY decimal(6,0) DEFAULT 8000,
MAX_SALARY decimal(6,0) DEFAULT NULL
);
```

Here is the structure of the table:

```
mysql> DESC jobs;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| JOB_ID     | varchar(10)  | NO   | PRI | NULL    |       |
| JOB_TITLE  | varchar(35)  | NO   |     |         |       |
| MIN_SALARY | decimal(6,0) | YES  |     | 8000    |       |
| MAX_SALARY | decimal(6,0) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

11. Write a SQL statement to create a table named countries including columns country_id, country_name and region_id and make sure that the country_id column will be a key field which will not contain any duplicate data at the time of insertion.

```
CREATE TABLE IF NOT EXISTS countries (
COUNTRY_ID varchar(2) NOT NULL UNIQUE PRIMARY KEY,
COUNTRY_NAME varchar(40) NOT NULL,
REGION_ID decimal(10,0) NOT NULL
);
```

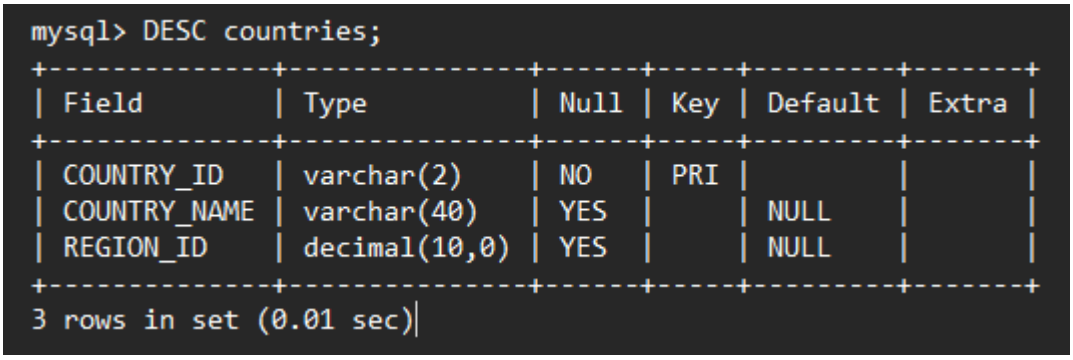
```
mysql> DESC countries;
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

12. Write a SQL statement to create a table countries including columns country_id, country_name and region_id and make sure that the column

country_id will be unique and store an auto incremented value.

```
CREATE TABLE IF NOT EXISTS countries (
COUNTRY_ID integer NOT NULL UNIQUE AUTO_INCREMENT PRIMARY KEY,
COUNTRY_NAME varchar(40) NOT NULL,
REGION_ID decimal(10,0) NOT NULL
);
DESC countries;
```

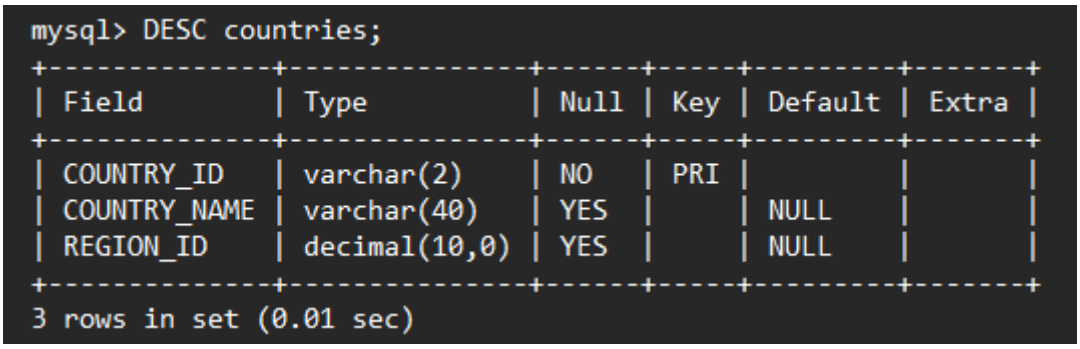
Here is the structure of the table:



13. Write a SQL statement to create a table countries including columns country_id, country_name and region_id and make sure that the combination of columns country_id and region_id will be unique.

```
CREATE TABLE IF NOT EXISTS countries (
COUNTRY_ID varchar(2) NOT NULL UNIQUE DEFAULT '',
COUNTRY_NAME varchar(40) DEFAULT NULL,
REGION_ID decimal(10,0) NOT NULL,
PRIMARY KEY (COUNTRY_ID,REGION_ID));
```

Here is the structure of the table:



14. Write a SQL statement to create a table job_history including columns employee_id, start_date, end_date, job_id and department_id and make sure that, the employee_id column does not contain any duplicate value at the time of insertion and the foreign key column job_id contain only those values which are exists in the jobs table.

Here is the structure of the table jobs;

```
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| JOB_ID     | varchar(10) | NO   | PRI |         |       |
| JOB_TITLE  | varchar(35) | NO   |     | NULL    |       |
| MIN_SALARY | decimal(6,0)| YES  |     | NULL    |       |
| MAX_SALARY | decimal(6,0)| YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
```

```
CREATE TABLE job_history (
EMPLOYEE_ID decimal(6,0) NOT NULL PRIMARY KEY,
START_DATE date NOT NULL,
END_DATE date NOT NULL,
JOB_ID varchar(10) NOT NULL,
DEPARTMENT_ID decimal(4,0) DEFAULT NULL,
FOREIGN KEY (job_id) REFERENCES jobs(job_id)
)ENGINE=InnoDB;
```

Here is the structure of the table:

```
mysql> DESC job_history;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| EMPLOYEE_ID   | decimal(6,0)| NO   | PRI | NULL    |       |
| START_DATE    | date        | NO   |     | NULL    |       |
| END_DATE      | date        | NO   |     | NULL    |       |
| JOB_ID        | varchar(10) | NO   | MUL | NULL    |       |
| DEPARTMENT_ID | decimal(4,0)| YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
5 rows in set (0.02 sec)
```

15. Write a SQL statement to create a table employees including columns employee_id, first_name, last_name, email, phone_number hire_date, job_id, salary, commission, manager_id and department_id and make sure that, the employee_id column does not contain any duplicate value at the time of insertion and the foreign key columns combined by department_id and

manager_id columns contain only those unique combination values, which combinations are exists in the departments table.

Assume the structure of departments table below.

```
+-----------------+--------------+------+-----+---------+-------+
| Field           | Type         | Null | Key | Default | Extra |
+-----------------+--------------+------+-----+---------+-------+
| DEPARTMENT_ID   | decimal(4,0) | NO   | PRI | 0       |       |
| DEPARTMENT_NAME | varchar(30)  | NO   |     | NULL    |       |
| MANAGER_ID      | decimal(6,0) | NO   | PRI | 0       |       |
| LOCATION_ID     | decimal(4,0) | YES  |     | NULL    |       |
+-----------------+--------------+------+-----+---------+-------+
```

```
CREATE TABLE IF NOT EXISTS employees (
EMPLOYEE_ID decimal(6,0) NOT NULL PRIMARY KEY,
FIRST_NAME varchar(20) DEFAULT NULL,
LAST_NAME varchar(25) NOT NULL,
EMAIL varchar(25) NOT NULL,
PHONE_NUMBER varchar(20) DEFAULT NULL,
HIRE_DATE date NOT NULL,
JOB_ID varchar(10) NOT NULL,
SALARY decimal(8,2) DEFAULT NULL,
COMMISSION_PCT decimal(2,2) DEFAULT NULL,
MANAGER_ID decimal(6,0) DEFAULT NULL,
DEPARTMENT_ID decimal(4,0) DEFAULT NULL,
FOREIGN KEY(DEPARTMENT_ID,MANAGER_ID)
REFERENCES  departments(DEPARTMENT_ID,MANAGER_ID)
)ENGINE=InnoDB;
```

Here is the structure of the table:

```
mysql> DESC employees;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| EMPLOYEE_ID    | decimal(6,0) | NO   | PRI | NULL    |       |
| FIRST_NAME     | varchar(20)  | YES  |     | NULL    |       |
| LAST_NAME      | varchar(25)  | NO   |     | NULL    |       |
| EMAIL          | varchar(25)  | NO   |     | NULL    |       |
| PHONE_NUMBER   | varchar(20)  | YES  |     | NULL    |       |
| HIRE_DATE      | date         | NO   |     | NULL    |       |
| JOB_ID         | varchar(10)  | NO   |     | NULL    |       |
| SALARY         | decimal(8,2) | YES  |     | NULL    |       |
| COMMISSION_PCT | decimal(2,2) | YES  |     | NULL    |       |
| MANAGER_ID     | decimal(6,0) | YES  |     | NULL    |       |
| DEPARTMENT_ID  | decimal(4,0) | YES  | MUL | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
11 rows in set (0.03 sec)
```

16. Write a SQL statement to create a table employees including columns employee_id, first_name, last_name, email, phone_number hire_date, job_id, salary, commission, manager_id and department_id and make sure that, the employee_id column does not contain any duplicate value at the time of insertion, and the foreign key column department_id, reference by the column department_id of departments table, can contain only those values which are exists in the departments table and another foreign key column job_id, referenced by the column job_id of jobs table, can contain only those values which are exists in the jobs table. The InnoDB Engine have been used to create the tables.

"A foreign key constraint is not required merely to join two tables. For storage engines other than InnoDB, it is possible when defining a column to use a REFERENCES tbl_name(col_name) clause, which has no actual effect, and serves only as a memo or comment to you that the column which you are currently defining is intended to refer to a column in another table." - Reference dev.mysql.com

Assume that the structure of two tables departments and jobs.

```
+-----------------+--------------+------+-----+---------+-------+
| Field           | Type         | Null | Key | Default | Extra |
+-----------------+--------------+------+-----+---------+-------+
| DEPARTMENT_ID   | decimal(4,0) | NO   | PRI | 0       |       |
| DEPARTMENT_NAME | varchar(30)  | NO   |     | NULL    |       |
| MANAGER_ID      | decimal(6,0) | YES  |     | NULL    |       |
```

```
| LOCATION_ID     | decimal(4,0) | YES  |     | NULL    |       |
+-----------------+--------------+------+-----+---------+-------+


+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| JOB_ID     | varchar(10)  | NO   | PRI |         |       |
| JOB_TITLE  | varchar(35)  | NO   |     | NULL    |       |
| MIN_SALARY | decimal(6,0) | YES  |     | NULL    |       |
| MAX_SALARY | decimal(6,0) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
```

```
CREATE TABLE IF NOT EXISTS employees (
EMPLOYEE_ID decimal(6,0) NOT NULL PRIMARY KEY,
FIRST_NAME varchar(20) DEFAULT NULL,
LAST_NAME varchar(25) NOT NULL,
EMAIL varchar(25) NOT NULL,
PHONE_NUMBER varchar(20) DEFAULT NULL,
HIRE_DATE date NOT NULL,
JOB_ID varchar(10) NOT NULL,
SALARY decimal(8,2) DEFAULT NULL,
COMMISSION_PCT decimal(2,2) DEFAULT NULL,
MANAGER_ID decimal(6,0) DEFAULT NULL,
DEPARTMENT_ID decimal(4,0) DEFAULT NULL,
FOREIGN KEY(DEPARTMENT_ID)
REFERENCES  departments(DEPARTMENT_ID),
FOREIGN KEY(JOB_ID)
REFERENCES  jobs(JOB_ID)
)ENGINE=InnoDB;
```

Here is the structure of the table:

```
mysql> DESC employees;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| EMPLOYEE_ID    | decimal(6,0) | NO   | PRI | NULL    |       |
| FIRST_NAME     | varchar(20)  | YES  |     | NULL    |       |
| LAST_NAME      | varchar(25)  | NO   |     | NULL    |       |
| EMAIL          | varchar(25)  | NO   |     | NULL    |       |
| PHONE_NUMBER   | varchar(20)  | YES  |     | NULL    |       |
| HIRE_DATE      | date         | NO   |     | NULL    |       |
| JOB_ID         | varchar(10)  | NO   |     | NULL    |       |
| SALARY         | decimal(8,2) | YES  |     | NULL    |       |
| COMMISSION_PCT | decimal(2,2) | YES  |     | NULL    |       |
| MANAGER_ID     | decimal(6,0) | YES  |     | NULL    |       |
| DEPARTMENT_ID  | decimal(4,0) | YES  | MUL | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
11 rows in set (0.01 sec)
```

17. Write a SQL statement to create a table employees including columns employee_id, first_name, last_name, job_id, salary and make sure that, the employee_id column does not contain any duplicate value at the time of insertion, and the foreign key column job_id, referenced by the column job_id of jobs table, can contain only those values which are exists in the jobs table. The InnoDB Engine have been used to create the tables. The specialty of the statement is that, The ON UPDATE CASCADE action allows you to perform cross-table update and ON DELETE RESTRICT action reject the deletion. The default action is ON DELETE RESTRICT.

Assume that the structure of the table jobs and InnoDB Engine have been used to create the table jobs.

```
CREATE TABLE IF NOT EXISTS jobs (
JOB_ID integer NOT NULL UNIQUE PRIMARY KEY,
JOB_TITLE varchar(35) NOT NULL DEFAULT ' ',
MIN_SALARY decimal(6,0) DEFAULT 8000,
MAX_SALARY decimal(6,0) DEFAULT NULL
)ENGINE=InnoDB;


+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| JOB_ID     | int(11)      | NO   | PRI | NULL    |       |
| JOB_TITLE  | varchar(35)  | NO   |     |         |       |
| MIN_SALARY | decimal(6,0) | YES  |     | 8000    |       |
| MAX_SALARY | decimal(6,0) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
```

```
CREATE TABLE IF NOT EXISTS employees (
EMPLOYEE_ID decimal(6,0) NOT NULL PRIMARY KEY,
FIRST_NAME varchar(20) DEFAULT NULL,
LAST_NAME varchar(25) NOT NULL,
EMAIL varchar(25) NOT NULL,
PHONE_NUMBER varchar(20) DEFAULT NULL,
HIRE_DATE date NOT NULL,
JOB_ID varchar(10) NOT NULL,
SALARY decimal(8,2) DEFAULT NULL,
COMMISSION_PCT decimal(2,2) DEFAULT NULL,
MANAGER_ID decimal(6,0) DEFAULT NULL,
DEPARTMENT_ID decimal(4,0) DEFAULT NULL,
FOREIGN KEY(DEPARTMENT_ID)
REFERENCES  departments(DEPARTMENT_ID),
FOREIGN KEY(JOB_ID)
REFERENCES  jobs(JOB_ID)
)ENGINE=InnoDB;
```

Here is the structure of the table:

```
mysql> DESC employees;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| EMPLOYEE_ID    | decimal(6,0) | NO   | PRI | NULL    |       |
| FIRST_NAME     | varchar(20)  | YES  |     | NULL    |       |
| LAST_NAME      | varchar(25)  | NO   |     | NULL    |       |
| EMAIL          | varchar(25)  | NO   |     | NULL    |       |
| PHONE_NUMBER   | varchar(20)  | YES  |     | NULL    |       |
| HIRE_DATE      | date         | NO   |     | NULL    |       |
| JOB_ID         | varchar(10)  | NO   |     | NULL    |       |
| SALARY         | decimal(8,2) | YES  |     | NULL    |       |
| COMMISSION_PCT | decimal(2,2) | YES  |     | NULL    |       |
| MANAGER_ID     | decimal(6,0) | YES  |     | NULL    |       |
| DEPARTMENT_ID  | decimal(4,0) | YES  | MUL | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
11 rows in set (0.01 sec)
```

18. Write a SQL statement to create a table employees including columns employee_id, first_name, last_name, job_id, salary and make sure that, the employee_id column does not contain any duplicate value at the time of insertion, and the foreign key column job_id, referenced by the column job_id of jobs table, can contain only those values which are exists in the jobs table. The InnoDB Engine have been used to create the tables. The specialty of the statement is that, The ON DELETE CASCADE that lets you allow to delete records in the employees(child) table that refer to a record in the jobs(parent) table when the record in the parent table is deleted and the ON UPDATE RESTRICT actions reject any updates.

Assume that the structure of the table jobs and InnoDB Engine have been used to create the table jobs.

```
CREATE TABLE IF NOT EXISTS jobs (
JOB_ID integer NOT NULL UNIQUE PRIMARY KEY,
JOB_TITLE varchar(35) NOT NULL DEFAULT ' ',
MIN_SALARY decimal(6,0) DEFAULT 8000,
MAX_SALARY decimal(6,0) DEFAULT NULL
)ENGINE=InnoDB;


+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| JOB_ID     | int(11)      | NO   | PRI | NULL    |       |
| JOB_TITLE  | varchar(35)  | NO   |     |         |       |
```

```
| MIN_SALARY | decimal(6,0) | YES  |     | 8000    |       |
| MAX_SALARY | decimal(6,0) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
```

```
CREATE TABLE IF NOT EXISTS employees (
EMPLOYEE_ID decimal(6,0) NOT NULL PRIMARY KEY,
FIRST_NAME varchar(20) DEFAULT NULL,
LAST_NAME varchar(25) NOT NULL,
JOB_ID INTEGER NOT NULL,
SALARY decimal(8,2) DEFAULT NULL,
FOREIGN KEY(JOB_ID)
REFERENCES  jobs(JOB_ID)
ON DELETE CASCADE ON UPDATE RESTRICT
)ENGINE=InnoDB;
```

Here is the structure of the table:

```
mysql> DESC employees;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| EMPLOYEE_ID    | decimal(6,0) | NO   | PRI | NULL    |       |
| FIRST_NAME     | varchar(20)  | YES  |     | NULL    |       |
| LAST_NAME      | varchar(25)  | NO   |     | NULL    |       |
| EMAIL          | varchar(25)  | NO   |     | NULL    |       |
| PHONE_NUMBER   | varchar(20)  | YES  |     | NULL    |       |
| HIRE_DATE      | date         | NO   |     | NULL    |       |
| JOB_ID         | varchar(10)  | NO   |     | NULL    |       |
| SALARY         | decimal(8,2) | YES  |     | NULL    |       |
| COMMISSION_PCT | decimal(2,2) | YES  |     | NULL    |       |
| MANAGER_ID     | decimal(6,0) | YES  |     | NULL    |       |
| DEPARTMENT_ID  | decimal(4,0) | YES  | MUL | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
11 rows in set (0.09 sec)
```

19. Write a SQL statement to create a table employees including columns employee_id, first_name, last_name, job_id, salary and make sure that, the employee_id column does not contain any duplicate value at the time of insertion, and the foreign key column job_id, referenced by the column job_id of jobs table, can contain only those values which are exists in the jobs table. The InnoDB Engine have been used to create the tables. The specialty of the statement is that, The ON DELETE SET NULL action will set the foreign key column values in the child table(employees) to NULL when the record in the parent table(jobs) is deleted, with a condition that the foreign key column in

the child table must accept NULL values and the ON UPDATE SET NULL action resets the values in the rows in the child table(employees) to NULL values when the rows in the parent table(jobs) are updated.

Assume that the structure of two table jobs and InnoDB Engine have been used to create the table jobs.

```
CREATE TABLE IF NOT EXISTS jobs (
JOB_ID integer NOT NULL UNIQUE PRIMARY KEY,
JOB_TITLE varchar(35) NOT NULL DEFAULT ' ',
MIN_SALARY decimal(6,0) DEFAULT 8000,
MAX_SALARY decimal(6,0) DEFAULT NULL
)ENGINE=InnoDB;


+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| JOB_ID     | int(11)      | NO   | PRI | NULL    |       |
| JOB_TITLE  | varchar(35)  | NO   |     |         |       |
| MIN_SALARY | decimal(6,0) | YES  |     | 8000    |       |
| MAX_SALARY | decimal(6,0) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
```

```
CREATE TABLE IF NOT EXISTS employees (
EMPLOYEE_ID decimal(6,0) NOT NULL PRIMARY KEY,
FIRST_NAME varchar(20) DEFAULT NULL,
LAST_NAME varchar(25) NOT NULL,
JOB_ID INTEGER,
SALARY decimal(8,2) DEFAULT NULL,
FOREIGN KEY(JOB_ID)
REFERENCES  jobs(JOB_ID)
ON DELETE SET NULL
ON UPDATE SET NULL
)ENGINE=InnoDB;
```

Here is the structure of the table:

```
mysql> DESC employees;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| EMPLOYEE_ID    | decimal(6,0) | NO   | PRI | NULL    |       |
| FIRST_NAME     | varchar(20)  | YES  |     | NULL    |       |
| LAST_NAME      | varchar(25)  | NO   |     | NULL    |       |
| EMAIL          | varchar(25)  | NO   |     | NULL    |       |
| PHONE_NUMBER   | varchar(20)  | YES  |     | NULL    |       |
| HIRE_DATE      | date         | NO   |     | NULL    |       |
| JOB_ID         | varchar(10)  | NO   |     | NULL    |       |
| SALARY         | decimal(8,2) | YES  |     | NULL    |       |
| COMMISSION_PCT | decimal(2,2) | YES  |     | NULL    |       |
| MANAGER_ID     | decimal(6,0) | YES  |     | NULL    |       |
| DEPARTMENT_ID  | decimal(4,0) | YES  | MUL | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
11 rows in set (0.01 sec)
```

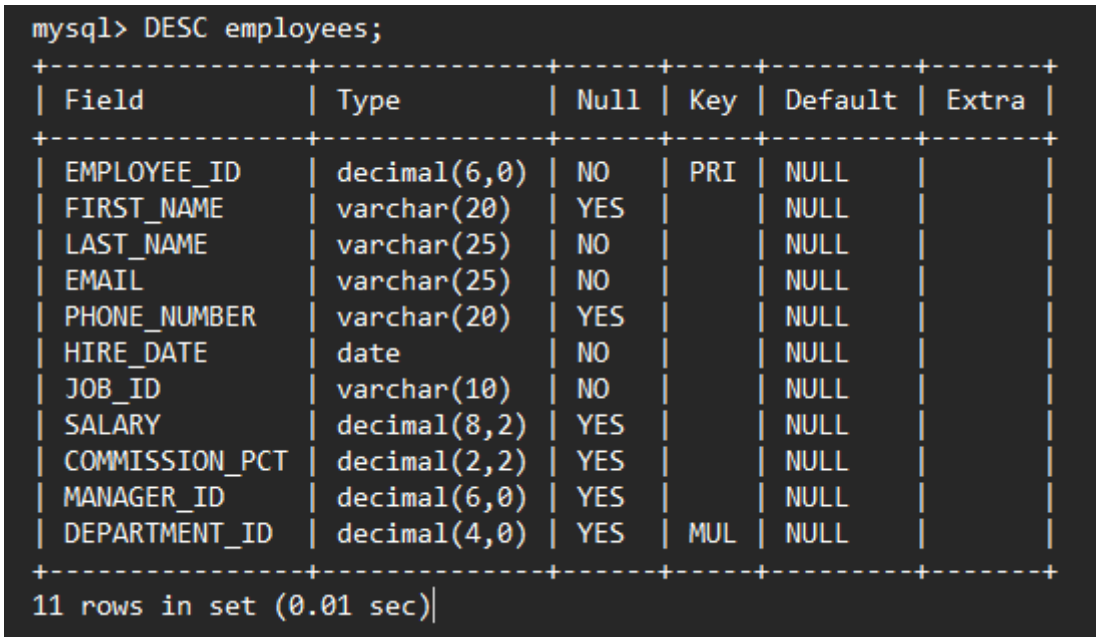
20. Write a SQL statement to create a table employees including columns employee_id, first_name, last_name, job_id, salary and make sure that, the employee_id column does not contain any duplicate value at the time of insertion, and the foreign key column job_id, referenced by the column job_id of jobs table, can contain only those values which are exists in the jobs table. The InnoDB Engine have been used to create the tables. The specialty of the statement is that, The ON DELETE NO ACTION and the ON UPDATE NO ACTION actions will reject the deletion and any updates.

Assume that the structure of two table jobs and InnoDB Engine have been used to create the table jobs.

```
CREATE TABLE IF NOT EXISTS jobs (
JOB_ID integer NOT NULL UNIQUE PRIMARY KEY,
JOB_TITLE varchar(35) NOT NULL DEFAULT ' ',
MIN_SALARY decimal(6,0) DEFAULT 8000,
MAX_SALARY decimal(6,0) DEFAULT NULL
)ENGINE=InnoDB;


+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| JOB_ID     | int(11)      | NO   | PRI | NULL    |       |
| JOB_TITLE  | varchar(35)  | NO   |     |         |       |
| MIN_SALARY | decimal(6,0) | YES  |     | 8000    |       |
| MAX_SALARY | decimal(6,0) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
```

```
CREATE TABLE IF NOT EXISTS employees (
EMPLOYEE_ID decimal(6,0) NOT NULL PRIMARY KEY,
FIRST_NAME varchar(20) DEFAULT NULL,
LAST_NAME varchar(25) NOT NULL,
JOB_ID INTEGER NOT NULL,
SALARY decimal(8,2) DEFAULT NULL,
FOREIGN KEY(JOB_ID)
REFERENCES  jobs(JOB_ID)
ON DELETE NO ACTION
ON UPDATE NO ACTION
)ENGINE=InnoDB;
```

Here is the structure of the table:



```
mysql> DESC employees;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| EMPLOYEE_ID    | decimal(6,0) | NO   | PRI | NULL    |       |
| FIRST_NAME     | varchar(20)  | YES  |     | NULL    |       |
| LAST_NAME      | varchar(25)  | NO   |     | NULL    |       |
| EMAIL          | varchar(25)  | NO   |     | NULL    |       |
| PHONE_NUMBER   | varchar(20)  | YES  |     | NULL    |       |
| HIRE_DATE      | date         | NO   |     | NULL    |       |
| JOB_ID         | varchar(10)  | NO   |     | NULL    |       |
| SALARY         | decimal(8,2) | YES  |     | NULL    |       |
| COMMISSION_PCT | decimal(2,2) | YES  |     | NULL    |       |
| MANAGER_ID     | decimal(6,0) | YES  |     | NULL    |       |
| DEPARTMENT_ID  | decimal(4,0) | YES  | MUL | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
11 rows in set (0.01 sec)
```

MySQL Insert Rows into the Table Questions

1. Write a SQL statement to insert a record with your own value into the table countries against each columns.

Here in the following is the structure of the table countries.

```
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
```

```
INSERT INTO countries VALUES('C1','India',1001);
```

Here is the structure of the table:

```
mysql> SELECT * FROM countries;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
| C1         | India        |      1001 |
+------------+--------------+-----------+
1 row in set (0.00 sec)
```

2. Write a SQL statement to insert one row into the table countries against the column country_id and country_name.

Here in the following is the structure of the table countries.

```
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
```

```
INSERT INTO countries (country_id,country_name) VALUES('C1','India');
```

Here is the structure of the table:

```
mysql> SELECT * FROM countries;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
| C1         | India        |      NULL |
+------------+--------------+-----------+
1 row in set (0.00 sec)
```

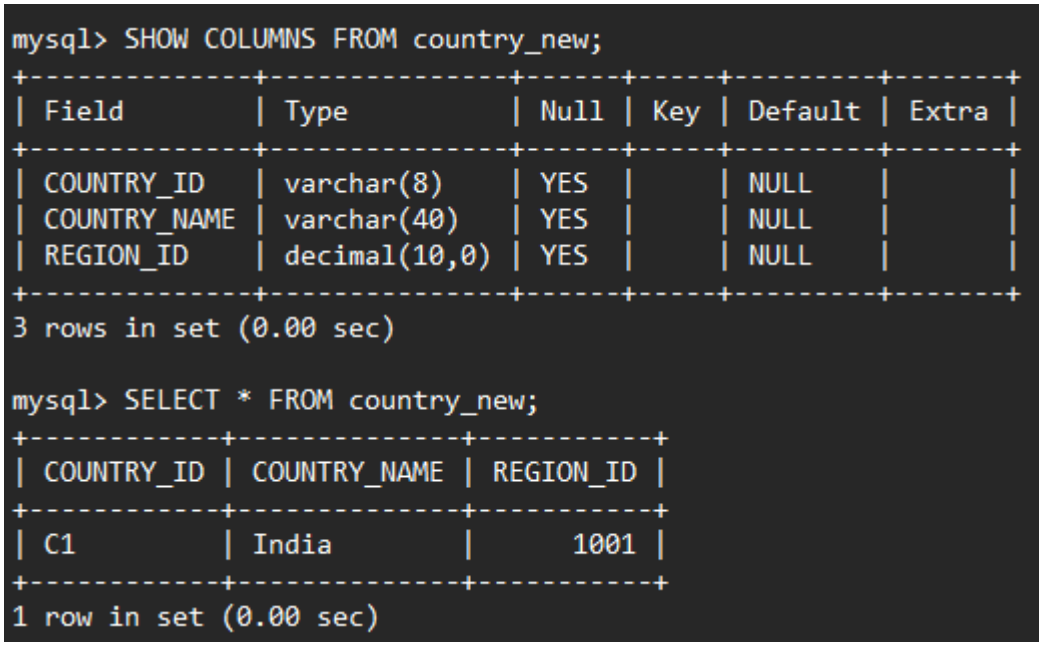
3. Write a SQL statement to create duplicate of countries table named country_new with all structure and data.

Here in the following is the structure of the table countries.

```
+--------------+---------------+------+-----+---------+-------+
| Field        | Type          | Null | Key | Default | Extra |
+--------------+---------------+------+-----+---------+-------+
| COUNTRY_ID   | varchar(2)    | YES  |     | NULL    |       |
| COUNTRY_NAME | varchar(40)   | YES  |     | NULL    |       |
| REGION_ID    | decimal(10,0) | YES  |     | NULL    |       |
+--------------+---------------+------+-----+---------+-------+
```

```
CREATE TABLE IF NOT EXISTS country_new
AS SELECT * FROM countries;
```

Here is the structure of the table:



4. Write a SQL statement to insert NULL values against region_id column for a row of countries table.

```
INSERT INTO countries (country_id,country_name,region_id) VALUES('C1','India',NULL);
```

Here is the structure of the table:

```
mysql> SELECT * FROM countries;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
| C1         | India        |      NULL |
+------------+--------------+-----------+
1 row in set (0.00 sec)
```

5. Write a SQL statement to insert 3 rows by a single insert statement.

```
INSERT INTO countries VALUES('C0001','India',1001),
('C0002','USA',1007),('C0003','UK',1003);
```

Here is the structure of the table:

```
mysql> SELECT * FROM COUNTRIES;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
| C0001      | India        |      1001 |
| C0002      | USA          |      1007 |
| C0003      | UK           |      1003 |
+------------+--------------+-----------+
3 rows in set (0.00 sec)
```

6. Write a SQL statement insert rows from country_new table to countries table.
Here is the rows for country_new table. Assume that, the countries table is empty.

```
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
| C0001      | India        |      1001 |
| C0002      | USA          |      1007 |
| C0003      | UK           |      1003 |
+------------+--------------+-----------+
```

```
INSERT INTO countries
SELECT * FROM country_new;
```

```
mysql> SELECT * FROM country_new;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
| C0001      | India        |      1001 |
| C0002      | USA          |      1007 |
| C0003      | UK           |      1003 |
+------------+--------------+-----------+
3 rows in set (0.00 sec)
```

7. Write a SQL statement to insert one row in jobs table to ensure that no duplicate value will be entered in the job_id column.

```
INSERT INTO jobs VALUES(1001,'OFFICER',8000);
```

```
mysql> INSERT INTO jobs VALUES(1001,'OFFICER',8000);
ERROR 1062 (23000): Duplicate entry '1001' for key 'JOB_ID'
```

8. Write a SQL statement to insert one row in jobs table to ensure that no duplicate value will be entered in the job_id column.

```
INSERT INTO jobs VALUES(1001,'OFFICER',8000);
```

```
mysql> INSERT INTO jobs VALUES(1001,'OFFICER',8000);
ERROR 1062 (23000): Duplicate entry '1001' for key 'PRIMARY'
```

9. Write a SQL statement to insert a record into the table countries to ensure that, a country_id and region_id combination will be entered once in the table.

```
INSERT INTO countries VALUES(501,'Italy',185);
```

```
mysql> INSERT INTO countries VALUES(501,'Italy',185);
ERROR 1062 (23000): Duplicate entry '501-185' for key 'PRIMARY'
```

10. Write a SQL statement to insert rows into the table countries in which the value of country_id column will be unique and auto incremented.

```
INSERT INTO countries(COUNTRY_NAME,REGION_ID) VALUES('India',185);
```

Here is the structure of the table:

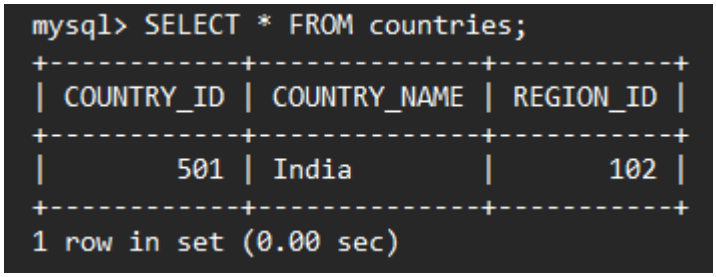
```
mysql> SELECT * FROM countries;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
|          1 | India        |       185 |
+------------+--------------+-----------+
1 row in set (0.00 sec)
```

```
INSERT INTO countries(COUNTRY_NAME,REGION_ID) VALUES('Japan',102);
```

```
mysql> SELECT * FROM countries;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
|          1 | India        |       185 |
|          2 | Japan        |       102 |
+------------+--------------+-----------+
2 rows in set (0.03 sec)
```

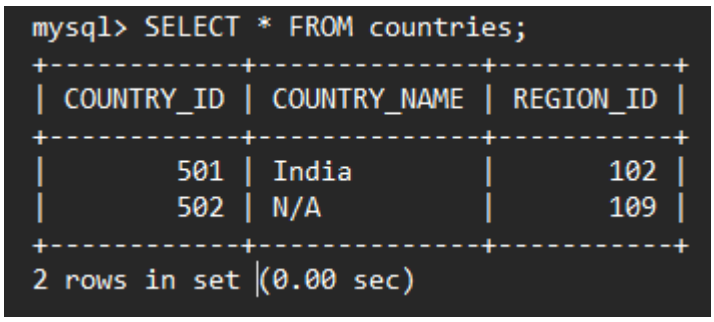
11. Write a SQL statement to insert records into the table countries to ensure that the country_id column will not contain any duplicate data and this will be automatically incremented and the column country_name will be filled up by 'N/A' if no value assigned for that column.

```
INSERT INTO countries VALUES(501,'India',102);
```



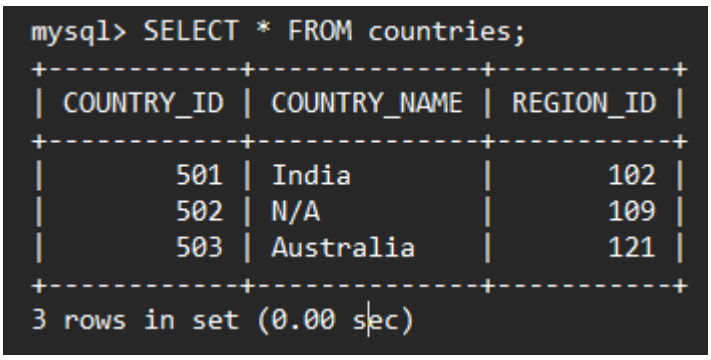
```
mysql> SELECT * FROM countries;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
|        501 | India        |       102 |
+------------+--------------+-----------+
1 row in set (0.00 sec)
```

```
INSERT INTO countries(region_id) VALUES(109);
```



```
mysql> SELECT * FROM countries;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
|        501 | India        |       102 |
|        502 | N/A          |       109 |
+------------+--------------+-----------+
2 rows in set (0.00 sec)
```

```
INSERT INTO countries(country_name,region_id) VALUES('Australia',121);
```



```
mysql> SELECT * FROM countries;
+------------+--------------+-----------+
| COUNTRY_ID | COUNTRY_NAME | REGION_ID |
+------------+--------------+-----------+
|        501 | India        |       102 |
|        502 | N/A          |       109 |
|        503 | Australia    |       121 |
+------------+--------------+-----------+
3 rows in set (0.00 sec)
```

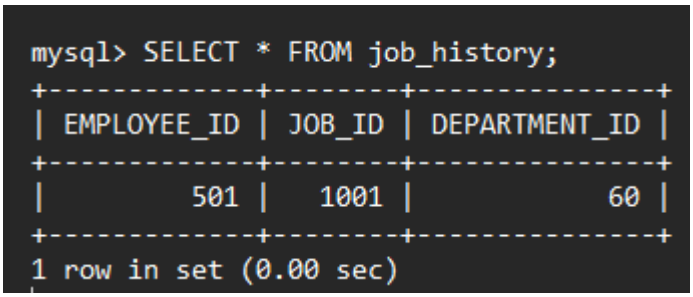
12. Write a SQL statement to insert rows in the job_history table in which one column job_id is containing those values which are exists in job_id column of jobs table.

```
CREATE TABLE IF NOT EXISTS jobs (
JOB_ID integer NOT NULL UNIQUE PRIMARY KEY,
JOB_TITLE varchar(35) NOT NULL DEFAULT ' ',
MIN_SALARY decimal(6,0) DEFAULT 8000,
MAX_SALARY decimal(6,0) DEFAULT 20000
)ENGINE=InnoDB;
```

```
INSERT INTO jobs(JOB_ID,JOB_TITLE) VALUES(1001,'OFFICER');
INSERT INTO jobs(JOB_ID,JOB_TITLE) VALUES(1002,'CLERK');

+--------+-----------+------------+------------+
| JOB_ID | JOB_TITLE | MIN_SALARY | MAX_SALARY |
+--------+-----------+------------+------------+
|   1001 | OFFICER   |       8000 |      20000 |
|   1002 | CLERK     |       8000 |      20000 |
+--------+-----------+------------+------------+
2 rows in set (0.00 sec)

Sample table job_history;

CREATE TABLE job_history (
EMPLOYEE_ID integer NOT NULL PRIMARY KEY,
JOB_ID integer NOT NULL,
DEPARTMENT_ID integer DEFAULT NULL,
FOREIGN KEY (job_id) REFERENCES jobs(job_id)
)ENGINE=InnoDB;
```

```
INSERT INTO job_history VALUES(501,1001,60);
```

Here is the structure of the table:



The value against job_id is 1001 which is exists in the job_id column of the jobs table, so no problem arise.

Now insert another row in the job_history table.

```
INSERT INTO job_history VALUES(502,1003,80);
```

Let execute the above code

```
mysql> INSERT INTO job_history VALUES(502,1003,80);
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails
(`hrr`.`job_history`, CONSTRAINT `job_history_ibfk_1`
 (`JOB_ID`) REFERENCES `jobs` (`JOB_ID`))
```

13. Write a SQL statement to insert rows into the table employees in which a set of columns department_id and manager_id contains a unique value and that combined values must have exists into the table departments.

```
Sample table departments.


CREATE TABLE IF NOT EXISTS departments (
DEPARTMENT_ID integer NOT NULL UNIQUE,
DEPARTMENT_NAME varchar(30) NOT NULL,
MANAGER_ID integer DEFAULT NULL,
LOCATION_ID integer DEFAULT NULL,
PRIMARY KEY (DEPARTMENT_ID)
)ENGINE=InnoDB;


INSERT INTO departments VALUES(60,'SALES',201,89);
INSERT INTO departments VALUES(61,'ACCOUNTS',201,89);


mysql> SELECT * FROM departments;
+---------------+-----------------+------------+-------------+
| DEPARTMENT_ID | DEPARTMENT_NAME | MANAGER_ID | LOCATION_ID |
+---------------+-----------------+------------+-------------+
|            60 | SALES           |        201 |          89 |
|            61 | ACCOUNTS        |        201 |          89 |
+---------------+-----------------+------------+-------------+
2 rows in set (0.00 sec)

Sample table jobs.

CREATE TABLE IF NOT EXISTS jobs (
JOB_ID integer NOT NULL UNIQUE PRIMARY KEY,
JOB_TITLE varchar(35) NOT NULL DEFAULT ' ',
MIN_SALARY decimal(6,0) DEFAULT 8000,
MAX_SALARY decimal(6,0) DEFAULT 20000
)ENGINE=InnoDB;


INSERT INTO jobs(JOB_ID,JOB_TITLE) VALUES(1001,'OFFICER');
INSERT INTO jobs(JOB_ID,JOB_TITLE) VALUES(1002,'CLERK');

mysql> SELECT * FROM jobs;
+--------+-----------+------------+------------+
| JOB_ID | JOB_TITLE | MIN_SALARY | MAX_SALARY |
+--------+-----------+------------+------------+
|   1001 | OFFICER   |       8000 |      20000 |
|   1002 | CLERK     |       8000 |      20000 |
+--------+-----------+------------+------------+
2 rows in set (0.00 sec)


Sample table employees.
```

```
CREATE TABLE IF NOT EXISTS employees (
EMPLOYEE_ID integer NOT NULL PRIMARY KEY,
FIRST_NAME varchar(20) DEFAULT NULL,
LAST_NAME varchar(25) NOT NULL,
DEPARTMENT_ID integer DEFAULT NULL,
FOREIGN KEY(DEPARTMENT_ID)
REFERENCES  departments(DEPARTMENT_ID),
JOB_ID integer NOT NULL,
FOREIGN KEY(JOB_ID)
REFERENCES  jobs(JOB_ID),
SALARY decimal(8,2) DEFAULT NULL
)ENGINE=InnoDB;
```

Now insert the rows into the table employees

```
INSERT INTO employees VALUES(510,'Alex','Hanes','CLERK',18000,201,60);
INSERT INTO employees VALUES(511,'Kim','Leon','CLERK',18000,211,80);
```

Here is the structure of the table:

```
mysql> SELECT * FROM employees;
+-------------+------------+-----------+--------+----------+------------+---------------+
| EMPLOYEE_ID | FIRST_NAME | LAST_NAME | JOB_ID | SALARY   | MANAGER_ID | DEPARTMENT_ID |
+-------------+------------+-----------+--------+----------+------------+---------------+
|         510 | Alex       | Hanes     | CLERK  | 18000.00 |        201 |            60 |
|         511 | Kim        | Leon      | CLERK  | 18000.00 |        211 |            80 |
+-------------+------------+-----------+--------+----------+------------+---------------+
2 rows in set (0.00 sec)
```

The value against department_id and manager_id combination (60,201) and (80,211) are unique in the departmentis(parent) table so, there is no problem arise to insert the rows in the child table employees.

Now insert another row in the employees table.

```
INSERT INTO employees VALUES(512,'Kim','Leon','CLERK',18000,80,211);
```

Let execute the above code

```
mysql> INSERT INTO employees VALUES(512,'Kim','Leon','CLERK',18000,80,211);
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails
(`hrr`.`employees`, CONSTRAINT `employees_ibfk_1` FOREIGN KEY (`D
EPARTMENT_ID`, `MANAGER_ID`) REFERENCES `departments` (`DEPARTMENT_ID`, `MANAGER_ID`))
```

Here in the above, the value against department_id and manager_id combination (211,80) does not matching with the same combination in departments(parent table) table and that is why the child table employees can not contain the combination of

values including department_id and manager_id as specified. Here the primary key - foreign key relationship is being violated and shows the above message.

14. Write a SQL statement to insert rows into the table employees in which a set of columns department_id and job_id contains the values which must have exists into the table departments and jobs.

```
Sample table departments.


CREATE TABLE IF NOT EXISTS departments (
DEPARTMENT_ID integer NOT NULL UNIQUE,
DEPARTMENT_NAME varchar(30) NOT NULL,
MANAGER_ID integer DEFAULT NULL,
LOCATION_ID integer DEFAULT NULL,
PRIMARY KEY (DEPARTMENT_ID)
)ENGINE=InnoDB;


INSERT INTO departments VALUES(60,'SALES',201,89);
INSERT INTO departments VALUES(61,'ACCOUNTS',201,89);


mysql> SELECT * FROM departments;
+---------------+-----------------+------------+-------------+
| DEPARTMENT_ID | DEPARTMENT_NAME | MANAGER_ID | LOCATION_ID |
+---------------+-----------------+------------+-------------+
|            60 | SALES           |        201 |          89 |
|            61 | ACCOUNTS        |        201 |          89 |
+---------------+-----------------+------------+-------------+
2 rows in set (0.00 sec)

Sample table jobs.

CREATE TABLE IF NOT EXISTS jobs (
JOB_ID integer NOT NULL UNIQUE PRIMARY KEY,
JOB_TITLE varchar(35) NOT NULL DEFAULT ' ',
MIN_SALARY decimal(6,0) DEFAULT 8000,
MAX_SALARY decimal(6,0) DEFAULT 20000
)ENGINE=InnoDB;


INSERT INTO jobs(JOB_ID,JOB_TITLE) VALUES(1001,'OFFICER');
INSERT INTO jobs(JOB_ID,JOB_TITLE) VALUES(1002,'CLERK');

mysql> SELECT * FROM jobs;
+--------+-----------+------------+------------+
| JOB_ID | JOB_TITLE | MIN_SALARY | MAX_SALARY |
+--------+-----------+------------+------------+
|   1001 | OFFICER   |       8000 |      20000 |
|   1002 | CLERK     |       8000 |      20000 |
+--------+-----------+------------+------------+
2 rows in set (0.00 sec)
```

```
Sample table employees.

CREATE TABLE IF NOT EXISTS employees (
EMPLOYEE_ID integer NOT NULL PRIMARY KEY,
FIRST_NAME varchar(20) DEFAULT NULL,
LAST_NAME varchar(25) NOT NULL,
DEPARTMENT_ID integer DEFAULT NULL,
FOREIGN KEY(DEPARTMENT_ID)
REFERENCES  departments(DEPARTMENT_ID),
JOB_ID integer NOT NULL,
FOREIGN KEY(JOB_ID)
REFERENCES  jobs(JOB_ID),
SALARY decimal(8,2) DEFAULT NULL
)ENGINE=InnoDB;
```

Now insert the rows into the table employees.

```
INSERT INTO employees VALUES(510,'Alex','Hanes',60,1001,18000);
```

Here is the structure of the table:

```
mysql> SELECT * FROM employees;
+-------------+------------+-----------+---------------+--------+----------+
| EMPLOYEE_ID | FIRST_NAME | LAST_NAME | DEPARTMENT_ID | JOB_ID | SALARY   |
+-------------+------------+-----------+---------------+--------+----------+
|         510 | Alex       | Hanes     |            60 |   1001 | 18000.00 |
+-------------+------------+-----------+---------------+--------+----------+
1 row in set (0.00 sec)
```

Here in the above insert statement the child column department_id and job_id of child table employees are successfully referencing with the department_id and job_id column of parent tables departments and jobs respectively, so no problem have been arisen to the insertion.

Now insert another row in the employees table.

```
INSERT INTO employees VALUES(511,'Tom','Elan',60,1003,22000);
```

Let execute the above code

```
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails
(`hrr`.`employees`, CONSTRAINT `employees_ibfk_2` FORE
OB_ID`) REFERENCES `jobs` (`JOB_ID`))
```

Here in the above insert statement show that, within child columns department_id and job_id of child table employees, the department_id are successfully referencing with the department_id of parent table departments but job_id column are not successfully referencing with the job_id of parent table jobs, so the problem have been arisen to the insertion displayed an error message.

Now insert another row in the employees table.

```
INSERT INTO employees VALUES(511,'Tom','Elan',80,1001,22000);
```

```
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails
(`hrr`.`employees`, CONSTRAINT `employees_ibfk_2` FOREIGN KEY (`J
OB_ID`) REFERENCES `jobs` (`JOB_ID`))
```

Here in the above insert statement show that, within child columns department_id and job_id of child table employees, the job_id are successfully referencing with the job_id of parent table jobs but department_id column are not successfully referencing with the department_id of parent table departments, so the problem have been arisen to the insertion and displayed the error message.

MySQL Update Table Questions

1. Write a SQL statement to change the email column of employees table with 'not available' for all employees.

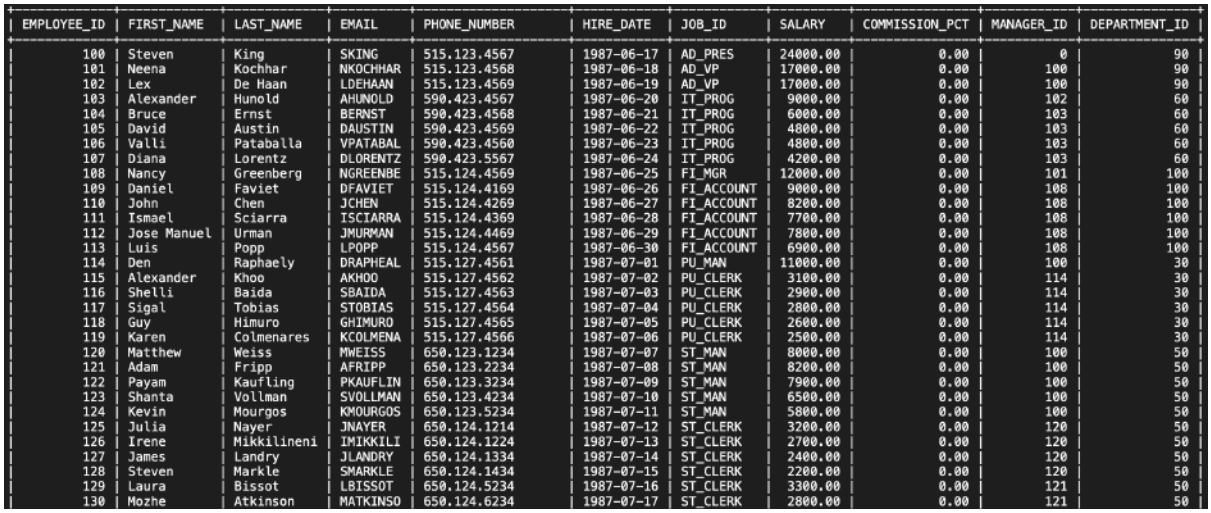
Here is the sample table employees.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

```
UPDATE employees SET email='not available';
```

2. Write a SQL statement to change the email and commission_pct column of employees table with 'not available' and 0.10 for all employees.

Here is the sample table employees.



EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

```
UPDATE employees SET email='not available',
commission_pct=0.10;
```

```
mysql> SELECT * FROM employees LIMIT 2;
+-------------+------------+-----------+---------------+--------------+------------+-----------+----------+----------------+------------+---------------+
| EMPLOYEE_ID | FIRST_NAME | LAST_NAME | EMAIL         | PHONE_NUMBER | HIRE_DATE  | JOB_ID    | SALARY   | COMMISSION_PCT | MANAGER_ID | DEPARTMENT_ID |
+-------------+------------+-----------+---------------+--------------+------------+-----------+----------+----------------+------------+---------------+
|         100 | Steven     | King      | not available | 515.123.4567 | 1987-06-17 | AD_PRES   | 24000.00 |           0.10 |          0 |            90 |
|         101 | Neena      | Kochhar   | not available | 515.123.4568 | 1987-06-18 | AD_VP     | 17000.00 |           0.10 |        100 |            90 |
+-------------+------------+-----------+---------------+--------------+------------+-----------+----------+----------------+------------+---------------+
2 rows in set (0.00 sec)
```

3. Write a SQL statement to change the email and commission_pct column of employees table with 'not available' and 0.10 for those employees whose department_id is 110.

Here is the sample table employees.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

```
UPDATE employees
SET email='not available',
commission_pct=0.10
WHERE department_id=110;
```

See the result. Only two rows have been displayed.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
205	Shelley	Higgins	not available	515.123.8080	1987-09-30	AC_MGR	12000.00	0.10	101	110
206	William	Gietz	not available	515.123.8181	1987-10-01	AC_ACCOUNT	8300.00	0.10	205	110

2 rows in set (0.00 sec)

4. Write a SQL statement to change the email column of employees table with 'not available' for those employees whose department_id is 80 and gets a commission is less than .20%

Here is the sample table employees.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

```
UPDATE employees
SET email='not available'
WHERE department_id=80 AND commission_pct<.20;
```

See the result. Only the effected rows have been displayed.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
155	Oliver	Tuvault	not available	011.44.1344.486508	1987-08-11	SA_REP	7000.00	0.15	145	80
163	Danielle	Greene	not available	011.44.1346.229268	1987-08-19	SA_REP	9500.00	0.15	147	80
164	Mattea	Marvins	not available	011.44.1346.329268	1987-08-20	SA_REP	7200.00	0.10	147	80
165	David	Lee	not available	011.44.1346.529268	1987-08-21	SA_REP	6800.00	0.10	147	80
166	Sundar	Ande	not available	011.44.1346.629268	1987-08-22	SA_REP	6400.00	0.10	147	80
167	Amit	Banda	not available	011.44.1346.729268	1987-08-23	SA_REP	6200.00	0.10	147	80
171	William	Smith	not available	011.44.1343.629268	1987-08-27	SA_REP	7400.00	0.15	148	80
172	Elizabeth	Bates	not available	011.44.1343.529268	1987-08-28	SA_REP	7300.00	0.15	148	80
173	Sundita	Kumar	not available	011.44.1343.329268	1987-08-29	SA_REP	6100.00	0.10	148	80
179	Charles	Johnson	not available	011.44.1644.429262	1987-09-04	SA_REP	6200.00	0.10	149	80

5. Write a SQL statement to change the email column of employees table with 'not available' for those employees who belongs to the 'Accouning' department.

Here is the sample table employees.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

Here is the sample table departments.

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing	114	1700
40	Human Resources	203	2400
50	Shipping	121	1500
60	IT	103	1400
70	Public Relations	204	2700

```
|             80 | Sales                |         145 |         2500 |
|             90 | Executive            |         100 |         1700 |
|            100 | Finance              |         108 |         1700 |
|            110 | Accounting           |         205 |         1700 |
|            120 | Treasury             |           0 |         1700 |
|            130 | Corporate Tax        |           0 |         1700 |
|            140 | Control And Credit   |           0 |         1700 |
|            150 | Shareholder Services |           0 |         1700 |
|            160 | Benefits             |           0 |         1700 |
|            170 | Manufacturing        |           0 |         1700 |
|            180 | Construction         |           0 |         1700 |
|            190 | Contracting          |           0 |         1700 |
|            200 | Operations           |           0 |         1700 |
|            210 | IT Support           |           0 |         1700 |
|            220 | NOC                  |           0 |         1700 |
|            230 | IT Helpdesk          |           0 |         1700 |
|            240 | Government Sales     |           0 |         1700 |
|            250 | Retail Sales         |           0 |         1700 |
|            260 | Recruiting           |           0 |         1700 |
|            270 | Payroll              |           0 |         1700 |
+---------------+----------------------+-------------+--------------+
```

```
UPDATE employees
SET email='not available'
WHERE department_id=(
SELECT department_id
FROM departments
WHERE department_name='Accounting');
```

See the result. Only the effected rows have been displayed.

```
+-------------+-----------+----------+---------------+--------------+-----------+------------+---------+----------------+------------+---------------+
| EMPLOYEE_ID | FIRST_NAME | LAST_NAME | EMAIL         | PHONE_NUMBER | HIRE_DATE | JOB_ID     | SALARY  | COMMISSION_PCT | MANAGER_ID | DEPARTMENT_ID |
+-------------+-----------+----------+---------------+--------------+-----------+------------+---------+----------------+------------+---------------+
|         205 | Shelley    | Higgins   | not available | 515.123.8080 | 1987-09-30 | AC_MGR     | 12000.00 |           0.00 |        101 |           110 |
|         206 | William    | Gietz     | not available | 515.123.8181 | 1987-10-01 | AC_ACCOUNT |  8300.00 |           0.00 |        205 |           110 |
+-------------+-----------+----------+---------------+--------------+-----------+------------+---------+----------------+------------+---------------+
```

6. Write a SQL statement to change salary of employee to 8000 whose ID is 105, if the existing salary is less than 5000.

Here is the sample table employees.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

```
UPDATE employees SET SALARY = 8000 WHERE employee_id = 105 AND salary < 5000;
```

See the result. Only the effected rows have been displayed.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	8000.00	0.00	103	60

7. Write a SQL statement to change job ID of employee which ID is 118, to SH_CLERK if the employee belongs to department, which ID is 30 and the existing job ID does not start with SH.

Here is the sample table employees.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

```
UPDATE employees SET JOB_ID= 'SH_CLERK'
WHERE employee_id=118
```

```
AND department_id=30
AND NOT JOB_ID LIKE 'SH%';
```

See the result. Only the effected rows have been displayed.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	SH_CLERK	2600.00	0.00	114	30

8. Write a SQL statement to increase the salary of employees under the department 40, 90 and 110 according to the company rules that, salary will be increased by 25% for the department 40, 15% for department 90 and 10% for the department 110 and the rest of the departments will remain same.

Here is the sample table employees.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

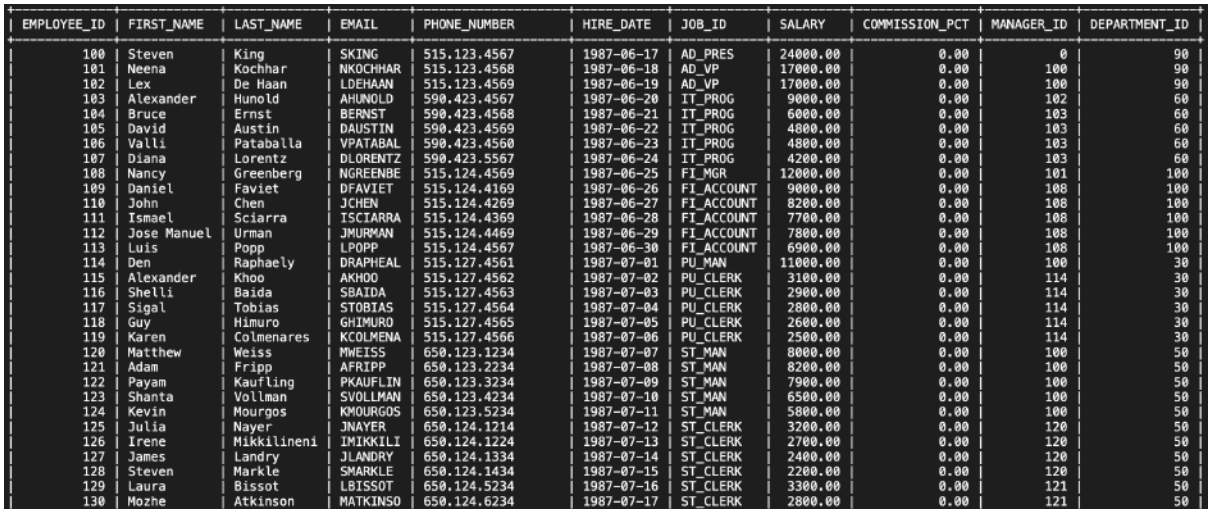
```
UPDATE employees SET salary= CASE department_id
WHEN 40 THEN salary+(salary*.25)
WHEN 90 THEN salary+(salary*.15)
WHEN 110 THEN salary+(salary*.10)
ELSE salary
END
WHERE department_id IN (40,50,50,60,70,80,90,110);
```

See the result before update. Only the effected rows have been displayed.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
203	Susan	Mavris	SMAVRIS	515.123.7777	1987-09-28	HR_REP	6500.00	0.00	101	40
205	Shelley	Higgins	SHIGGINS	515.123.8080	1987-09-30	AC_MGR	12000.00	0.00	101	110
206	William	Gietz	WGIETZ	515.123.8181	1987-10-01	AC_ACCOUNT	8300.00	0.00	205	110

9. Write a SQL statement to increase the minimum and maximum salary of PU_CLERK by 2000 as well as the salary for those employees by 20% and commission percent by .10.

Here is the sample table employees.



EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50

Here is the sample table jobs.

```
+------------+----------------------------------+------------+------------+
| JOB_ID     | JOB_TITLE                        | MIN_SALARY | MAX_SALARY |
+------------+----------------------------------+------------+------------+
| AD_PRES    | President                        |      20000 |      40000 |
| AD_VP      | Administration Vice President    |      15000 |      30000 |
| AD_ASST    | Administration Assistant         |       3000 |       6000 |
| FI_MGR     | Finance Manager                  |       8200 |      16000 |
| FI_ACCOUNT | Accountant                       |       4200 |       9000 |
| AC_MGR     | Accounting Manager               |       8200 |      16000 |
| AC_ACCOUNT | Public Accountant                |       4200 |       9000 |
| SA_MAN     | Sales Manager                    |      10000 |      20000 |
| SA_REP     | Sales Representative             |       6000 |      12000 |
| PU_MAN     | Purchasing Manager               |       8000 |      15000 |
| PU_CLERK   | Purchasing Clerk                 |       2500 |       5500 |
| ST_MAN     | Stock Manager                    |       5500 |       8500 |
| ST_CLERK   | Stock Clerk                      |       2000 |       5000 |
| SH_CLERK   | Shipping Clerk                   |       2500 |       5500 |
| IT_PROG    | Programmer                       |       4000 |      10000 |
| MK_MAN     | Marketing Manager                |       9000 |      15000 |
| MK_REP     | Marketing Representative         |       4000 |       9000 |
| HR_REP     | Human Resources Representative   |       4000 |       9000 |
| PR_REP     | Public Relations Representative  |       4500 |      10500 |
+------------+----------------------------------+------------+------------+
```

```
UPDATE jobs,employees
SET jobs.min_salary=jobs.min_salary+2000,
```

```
jobs.max_salary=jobs.max_salary+2000,
employees.salary=employees.salary+(employees.salary*.20),
employees.commission_pct=employees.commission_pct+.10
WHERE jobs.job_id='PU_CLERK'
AND employees.job_id='PU_CLERK';
```

See the result before update. Only the effected rows have been displayed.

```
table -  jobs
+----------+-----------------+------------+------------+
| JOB_ID   | JOB_TITLE       | MIN_SALARY | MAX_SALARY |
+----------+-----------------+------------+------------+
| PU_CLERK | Purchasing Clerk |       2500 |       5500 |
+----------+-----------------+------------+------------+


table - employees
+-------------+------------+------------+----------+-------------+-----------+----------+---------+---------------+------------+---------------+
| EMPLOYEE_ID | FIRST_NAME | LAST_NAME  | EMAIL    | PHONE_NUMBER | HIRE_DATE | JOB_ID   | SALARY  | COMMISSION_PCT | MANAGER_ID | DEPARTMENT_ID |
+-------------+------------+------------+----------+-------------+-----------+----------+---------+---------------+------------+---------------+
|         115 | Alexander  | Khoo       | AKHOO    | 515.127.4562 | 1987-07-02 | PU_CLERK | 3100.00 |          0.00 |        114 |            30 |
|         116 | Shelli     | Baida      | SBAIDA   | 515.127.4563 | 1987-07-03 | PU_CLERK | 2900.00 |          0.00 |        114 |            30 |
|         117 | Sigal      | Tobias     | STOBIAS  | 515.127.4564 | 1987-07-04 | PU_CLERK | 2800.00 |          0.00 |        114 |            30 |
|         118 | Guy        | Himuro     | GHIMURO  | 515.127.4565 | 1987-07-05 | PU_CLERK | 2600.00 |          0.00 |        114 |            30 |
|         119 | Karen      | Colmenares | KCOLMENA | 515.127.4566 | 1987-07-06 | PU_CLERK | 2500.00 |          0.00 |        114 |            30 |
+-------------+------------+------------+----------+-------------+-----------+----------+---------+---------------+------------+---------------+
```

See the result. Only the effected rows have been displayed.

```
table - jobs
+----------+-----------------+------------+------------+
| JOB_ID   | JOB_TITLE       | MIN_SALARY | MAX_SALARY |
+----------+-----------------+------------+------------+
| PU_CLERK | Purchasing Clerk |       4500 |       7500 |
+----------+-----------------+------------+------------+


table - employees
+-------------+------------+------------+----------+-------------+-----------+----------+---------+---------------+------------+---------------+
| EMPLOYEE_ID | FIRST_NAME | LAST_NAME  | EMAIL    | PHONE_NUMBER | HIRE_DATE | JOB_ID   | SALARY  | COMMISSION_PCT | MANAGER_ID | DEPARTMENT_ID |
+-------------+------------+------------+----------+-------------+-----------+----------+---------+---------------+------------+---------------+
|         115 | Alexander  | Khoo       | AKHOO    | 515.127.4562 | 1987-07-02 | PU_CLERK | 3720.00 |          0.10 |        114 |            30 |
|         116 | Shelli     | Baida      | SBAIDA   | 515.127.4563 | 1987-07-03 | PU_CLERK | 3480.00 |          0.10 |        114 |            30 |
|         117 | Sigal      | Tobias     | STOBIAS  | 515.127.4564 | 1987-07-04 | PU_CLERK | 3360.00 |          0.10 |        114 |            30 |
|         118 | Guy        | Himuro     | GHIMURO  | 515.127.4565 | 1987-07-05 | PU_CLERK | 3120.00 |          0.10 |        114 |            30 |
|         119 | Karen      | Colmenares | KCOLMENA | 515.127.4566 | 1987-07-06 | PU_CLERK | 3000.00 |          0.10 |        114 |            30 |
+-------------+------------+------------+----------+-------------+-----------+----------+---------+---------------+------------+---------------+
```

Here is the FULL sample table employees.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1987-06-21	IT_PROG	6000.00	0.00	103	60
105	David	Austin	DAUSTIN	590.423.4569	1987-06-22	IT_PROG	4800.00	0.00	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1987-06-23	IT_PROG	4800.00	0.00	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	1987-06-24	IT_PROG	4200.00	0.00	103	60
108	Nancy	Greenberg	NGREENBE	515.124.4569	1987-06-25	FI_MGR	12000.00	0.00	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1987-06-26	FI_ACCOUNT	9000.00	0.00	108	100
110	John	Chen	JCHEN	515.124.4269	1987-06-27	FI_ACCOUNT	8200.00	0.00	108	100
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1987-06-28	FI_ACCOUNT	7700.00	0.00	108	100
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1987-06-29	FI_ACCOUNT	7800.00	0.00	108	100
113	Luis	Popp	LPOPP	515.124.4567	1987-06-30	FI_ACCOUNT	6900.00	0.00	108	100
114	Den	Raphaely	DRAPHEAL	515.127.4561	1987-07-01	PU_MAN	11000.00	0.00	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1987-07-02	PU_CLERK	3100.00	0.00	114	30
116	Shelli	Baida	SBAIDA	515.127.4563	1987-07-03	PU_CLERK	2900.00	0.00	114	30
117	Sigal	Tobias	STOBIAS	515.127.4564	1987-07-04	PU_CLERK	2800.00	0.00	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1987-07-05	PU_CLERK	2600.00	0.00	114	30
119	Karen	Colmenares	KCOLMENA	515.127.4566	1987-07-06	PU_CLERK	2500.00	0.00	114	30
120	Matthew	Weiss	MWEISS	650.123.1234	1987-07-07	ST_MAN	8000.00	0.00	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1987-07-08	ST_MAN	8200.00	0.00	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1987-07-09	ST_MAN	7900.00	0.00	100	50
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1987-07-10	ST_MAN	6500.00	0.00	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1987-07-11	ST_MAN	5800.00	0.00	100	50
125	Julia	Nayer	JNAYER	650.124.1214	1987-07-12	ST_CLERK	3200.00	0.00	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1987-07-13	ST_CLERK	2700.00	0.00	120	50
127	James	Landry	JLANDRY	650.124.1334	1987-07-14	ST_CLERK	2400.00	0.00	120	50
128	Steven	Markle	SMARKLE	650.124.1434	1987-07-15	ST_CLERK	2200.00	0.00	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1987-07-16	ST_CLERK	3300.00	0.00	121	50
130	Mozhe	Atkinson	MATKINSO	650.124.6234	1987-07-17	ST_CLERK	2800.00	0.00	121	50
131	James	Marlow	JAMRLOW	650.124.7234	1987-07-18	ST_CLERK	2500.00	0.00	121	50
132	TJ	Olson	TJOLSON	650.124.8234	1987-07-19	ST_CLERK	2100.00	0.00	121	50
133	Jason	Mallin	JMALLIN	650.127.1934	1987-07-20	ST_CLERK	3300.00	0.00	122	50
134	Michael	Rogers	MROGERS	650.127.1834	1987-07-21	ST_CLERK	2900.00	0.00	122	50
135	Ki	Gee	KGEE	650.127.1734	1987-07-22	ST_CLERK	2400.00	0.00	122	50
136	Hazel	Philtanker	HPHILTAN	650.127.1634	1987-07-23	ST_CLERK	2200.00	0.00	122	50
137	Renske	Ladwig	RLADWIG	650.121.1234	1987-07-24	ST_CLERK	3600.00	0.00	123	50
138	Stephen	Stiles	SSTILES	650.121.2034	1987-07-25	ST_CLERK	3200.00	0.00	123	50
139	John	Seo	JSEO	650.121.2019	1987-07-26	ST_CLERK	2700.00	0.00	123	50
140	Joshua	Patel	JPATEL	650.121.1834	1987-07-27	ST_CLERK	2500.00	0.00	123	50
141	Trenna	Rajs	TRAJS	650.121.8009	1987-07-28	ST_CLERK	3500.00	0.00	124	50
142	Curtis	Davies	CDAVIES	650.121.2994	1987-07-29	ST_CLERK	3100.00	0.00	124	50
143	Randall	Matos	RMATOS	650.121.2874	1987-07-30	ST_CLERK	2600.00	0.00	124	50
144	Peter	Vargas	PVARGAS	650.121.2004	1987-07-31	ST_CLERK	2500.00	0.00	124	50
145	John	Russell	JRUSSEL	011.44.1344.429268	1987-08-01	SA_MAN	14000.00	0.40	100	80
146	Karen	Partners	KPARTNER	011.44.1344.467268	1987-08-02	SA_MAN	13500.00	0.30	100	80
147	Alberto	Errazuriz	AERRAZUR	011.44.1344.429278	1987-08-03	SA_MAN	12000.00	0.30	100	80
148	Gerald	Cambrault	GCAMBRAU	011.44.1344.619268	1987-08-04	SA_MAN	11000.00	0.30	100	80
149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	1987-08-05	SA_MAN	10500.00	0.20	100	80
150	Peter	Tucker	PTUCKER	011.44.1344.129268	1987-08-06	SA_REP	10000.00	0.30	145	80
151	David	Bernstein	DBERNSTE	011.44.1344.345268	1987-08-07	SA_REP	9500.00	0.25	145	80
152	Peter	Hall	PHALL	011.44.1344.478968	1987-08-08	SA_REP	9000.00	0.25	145	80
153	Christopher	Olsen	COLSEN	011.44.1344.498718	1987-08-09	SA_REP	8000.00	0.20	145	80
154	Nanette	Cambrault	NCAMBRAU	011.44.1344.987668	1987-08-10	SA_REP	7500.00	0.20	145	80
155	Oliver	Tuvault	OTUVAULT	011.44.1344.486508	1987-08-11	SA_REP	7000.00	0.15	145	80
156	Janette	King	JKING	011.44.1345.429268	1987-08-12	SA_REP	10000.00	0.35	146	80
157	Patrick	Sully	PSULLY	011.44.1345.929268	1987-08-13	SA_REP	9500.00	0.35	146	80
158	Allan	McEwen	AMCEWEN	011.44.1345.829268	1987-08-14	SA_REP	9000.00	0.35	146	80
159	Lindsey	Smith	LSMITH	011.44.1345.729268	1987-08-15	SA_REP	8000.00	0.30	146	80
160	Louise	Doran	LDORAN	011.44.1345.629268	1987-08-16	SA_REP	7500.00	0.30	146	80
161	Sarath	Sewall	SSEWALL	011.44.1345.529268	1987-08-17	SA_REP	7000.00	0.25	146	80
162	Clara	Vishney	CVISHNEY	011.44.1346.129268	1987-08-18	SA_REP	10500.00	0.25	147	80
163	Danielle	Greene	DGREENE	011.44.1346.229268	1987-08-19	SA_REP	9500.00	0.15	147	80
164	Mattea	Marvins	MMARVINS	011.44.1346.329268	1987-08-20	SA_REP	7200.00	0.10	147	80
165	David	Lee	DLEE	011.44.1346.529268	1987-08-21	SA_REP	6800.00	0.10	147	80
166	Sundar	Ande	SANDE	011.44.1346.629268	1987-08-22	SA_REP	6400.00	0.10	147	80
167	Amit	Banda	ABANDA	011.44.1346.729268	1987-08-23	SA_REP	6200.00	0.10	147	80
168	Lisa	Ozer	LOZER	011.44.1343.929268	1987-08-24	SA_REP	11500.00	0.25	148	80
169	Harrison	Bloom	HBLOOM	011.44.1343.829268	1987-08-25	SA_REP	10000.00	0.20	148	80
170	Tayler	Fox	TFOX	011.44.1343.729268	1987-08-26	SA_REP	9600.00	0.20	148	80
171	William	Smith	WSMITH	011.44.1343.629268	1987-08-27	SA_REP	7400.00	0.15	148	80
172	Elizabeth	Bates	EBATES	011.44.1343.529268	1987-08-28	SA_REP	7300.00	0.15	148	80
173	Sundita	Kumar	SKUMAR	011.44.1343.329268	1987-08-29	SA_REP	6100.00	0.10	148	80
174	Ellen	Abel	EABEL	011.44.1644.429267	1987-08-30	SA_REP	11000.00	0.30	149	80
175	Alyssa	Hutton	AHUTTON	011.44.1644.429266	1987-08-31	SA_REP	8800.00	0.25	149	80
176	Jonathon	Taylor	JTAYLOR	011.44.1644.429265	1987-09-01	SA_REP	8600.00	0.20	149	80
177	Jack	Livingston	JLIVINGS	011.44.1644.429264	1987-09-02	SA_REP	8400.00	0.20	149	80
178	Kimberely	Grant	KGRANT	011.44.1644.429263	1987-09-03	SA_REP	7000.00	0.15	149	0
179	Charles	Johnson	CJOHNSON	011.44.1644.429262	1987-09-04	SA_REP	6200.00	0.10	149	80
180	Winston	Taylor	WTAYLOR	650.507.9876	1987-09-05	SH_CLERK	3200.00	0.00	120	50
181	Jean	Fleaur	JFLEAUR	650.507.9877	1987-09-06	SH_CLERK	3100.00	0.00	120	50
182	Martha	Sullivan	MSULLIVA	650.507.9878	1987-09-07	SH_CLERK	2500.00	0.00	120	50
183	Girard	Geoni	GGEONI	650.507.9879	1987-09-08	SH_CLERK	2800.00	0.00	120	50
184	Nandita	Sarchand	NSARCHAN	650.509.1876	1987-09-09	SH_CLERK	4200.00	0.00	121	50
185	Alexis	Bull	ABULL	650.509.2876	1987-09-10	SH_CLERK	4100.00	0.00	121	50
186	Julia	Dellinger	JDELLING	650.509.3876	1987-09-11	SH_CLERK	3400.00	0.00	121	50
187	Anthony	Cabrio	ACABRIO	650.509.4876	1987-09-12	SH_CLERK	3000.00	0.00	121	50
188	Kelly	Chung	KCHUNG	650.505.1876	1987-09-13	SH_CLERK	3800.00	0.00	122	50
189	Jennifer	Dilly	JDILLY	650.505.2876	1987-09-14	SH_CLERK	3600.00	0.00	122	50
190	Timothy	Gates	TGATES	650.505.3876	1987-09-15	SH_CLERK	2900.00	0.00	122	50
191	Randall	Perkins	RPERKINS	650.505.4876	1987-09-16	SH_CLERK	2500.00	0.00	122	50
192	Sarah	Bell	SBELL	650.501.1876	1987-09-17	SH_CLERK	4000.00	0.00	123	50
193	Britney	Everett	BEVERETT	650.501.2876	1987-09-18	SH_CLERK	3900.00	0.00	123	50
194	Samuel	McCain	SMCCAIN	650.501.3876	1987-09-19	SH_CLERK	3200.00	0.00	123	50
195	Vance	Jones	VJONES	650.501.4876	1987-09-20	SH_CLERK	2800.00	0.00	123	50
196	Alana	Walsh	AWALSH	650.507.9811	1987-09-21	SH_CLERK	3100.00	0.00	124	50
197	Kevin	Feeney	KFEENEY	650.507.9822	1987-09-22	SH_CLERK	3000.00	0.00	124	50
198	Donald	OConnell	DOCONNEL	650.507.9833	1987-09-23	SH_CLERK	2600.00	0.00	124	50
199	Douglas	Grant	DGRANT	650.507.9844	1987-09-24	SH_CLERK	2600.00	0.00	124	50
200	Jennifer	Whalen	JWHALEN	515.123.4444	1987-09-25	AD_ASST	4400.00	0.00	101	10
201	Michael	Hartstein	MHARTSTE	515.123.5555	1987-09-26	MK_MAN	13000.00	0.00	100	20
202	Pat	Fay	PFAY	603.123.6666	1987-09-27	MK_REP	6000.00	0.00	201	20
203	Susan	Mavris	SMAVRIS	515.123.7777	1987-09-28	HR_REP	6500.00	0.00	101	40
204	Hermann	Baer	HBAER	515.123.8888	1987-09-29	PR_REP	10000.00	0.00	101	70
205	Shelley	Higgins	SHIGGINS	515.123.8080	1987-09-30	AC_MGR	12000.00	0.00	101	110
206	William	Gietz	WGIETZ	515.123.8181	1987-10-01	AC_ACCOUNT	8300.00	0.00	205	110