SQL Queries

1. Create/Delete/Edit/Display an Employer.

From the Plans table, we see that the plan with ID 5 has userType ‘employer’

planID	name	price	applyLimit	postLimit	userType
1	Employee Basic	0	0	0	employee
2	Employee Prime	10	5	0	employee
3	Employee Gold	20	NULL	0	employee
4	Employer Prime	50	0	5	employer
5	Employer Gold	100	0	NULL	employer
6	Admin	0	0	0	admin
NULL	NULL	NULL	NULL	NULL	NULL

CREATE

```
INSERT INTO Users (userID,
                   planID,
                   email,
                   password,
                   isActive,
                   balance,
                   isAutomatic)
VALUES     (‘bob’,
                    4,
                    'bob@comp353.com',
                    'bob',
                    TRUE,
                    0,
                    TRUE);
```

userNumber	userID	planID	email	password	dateCreated	isActive	startSufferingDate	balance	isAutomatic
6	bob	5	bob@comp353.com	bob	2020-05-15 00:00:00	1	NULL	0	0

EDIT

```
UPDATE Users
SET isActive = 0
WHERE userNumber=6;
```

DISPLAY

```
SELECT * FROM Users
WHERE userNumber=6;
```

DELETE
DELETE FROM Users
WHERE userNumber=6;

Note: This delete will cascade and delete rows from all the other tables where the userID ‘bob’ is a FK.

2. **Create/Delete/Edit/Display a category by an Employer.**

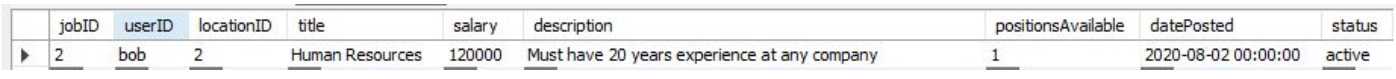
Create
INSERT INTO Employer_Categories (userID, categoryName)
VALUES ('alice', 'Software Engineer'),
('alice', 'Tech Lead');
EDIT
UPDATE Employer_Categories
SET categoryName = 'Project Manager'
WHERE categoryName = 'Tech Lead'
AND userID = ‘alice’

DISPLAY
SELECT * FROM Employer_Categories
WHERE categoryName = 'Project Manager'
AND userID = ‘alice’

DELETE
DELETE FROM Employer_Categories
WHERE categoryName = 'Project Manager'
AND userID = 'alice'

3. **Post a new job by an employer.**

INSERT INTO Jobs
(userID, locationID, title, salary, description, positionsAvailable, status)
VALUES ('bob', 2, 'Human Resources', 120000, 'Must have 20 years of experience at any company',1, ‘active’);



jobID	userID	locationID	title	salary	description	positionsAvailable	datePosted	status
2	bob	2	Human Resources	120000	Must have 20 years experience at any company	1	2020-08-02 00:00:00	active

4. **Provide a job offer for an employee by an employer.**

Applications table before (user has applied to a job):

jobID	userID	dateApplied	isAcceptedByEmployer	isAcceptedByEmployee
3	alice	2020-08-01 19:31:17	NULL	NULL

Query:

```
UPDATE Applications SET isAcceptedByEmployer = 1 WHERE userID = 'alice' AND jobID = 3;
```

Application after query (employer has accepted the application and offered a job):

jobID	userID	dateApplied	isAcceptedByEmployer	isAcceptedByEmployee
3	alice	2020-08-01 19:31:17	1	NULL

5. **Report of a posted job by an employer (Job title and description, date posted, list of employees applied to the job and status of each application).**

```
SELECT j.title, j.description, j.datePosted, IFNULL(a.userID, 'No Applications') AS 'username',
IFNULL(a.isAcceptedByEmployer, 'N/A') AS 'isAcceptedByEmployer',
IFNULL(a.isAcceptedByEmployee, 'N/A') AS 'isAccemptedByEmployee'
FROM Jobs AS j
LEFT JOIN Applications AS a ON j.jobID = a.jobID
WHERE j.jobID = [input.jobID]
```

title	description	datePosted	username	isAcceptedByEmployer	isAccemptedByEmployee
Human Resources	Must have 20 years experience at any company	2020-08-02 00:00:00	tiffany	N/A	N/A
Human Resources	Must have 20 years experience at any company	2020-08-02 00:00:00	gordon	N/A	N/A
Human Resources	Must have 20 years experience at any company	2020-08-02 00:00:00	simba	N/A	N/A
Human Resources	Must have 20 years experience at any company	2020-08-02 00:00:00	arun	N/A	N/A
Human Resources	Must have 20 years experience at any company	2020-08-02 00:00:00	tyson	1	1

6. **Report of posted jobs by an employer during a specific period of time (Job title, date posted, short description of the job up to 50 characters, number of needed employees to the post, number of applied jobs to the post, number of accepted offers).**

```
SELECT j.title, SUBSTRING(j.description, 1, 50) AS 'description', j.positionsAvailable,
j.datePosted, COUNT(a.jobID) AS 'Number of employees applied',
SUM(IFNULL(a.isAcceptedByEmployee, 0)) AS 'Number of accepted offers'
FROM Jobs AS j
LEFT JOIN Applications AS a ON j.jobID = a.jobID
WHERE j.userID = [input_userID] AND
j.datePosted BETWEEN 'YYYY-MM-DD' AND 'YYYY-MM-DD'
GROUP BY j.jobID;
```

title	description	positionsAvailable	datePosted	Number of employees applied	Number of accepted offers
Python Developer	Must know snakes	5	2020-08-01 00:00:00	1	0
Java Developer	Must like coffee	2	2020-08-02 00:00:00	1	0
JavaScript Developer	Must know all frameworks	8	2020-08-03 00:00:00	0	0
Rust Developer	We will ask about metals	3	2020-08-04 00:00:00	0	0
Software Developer	Work on Angular apps	4	2020-08-06 21:40:38	0	0

7. **Create/Delete/Edit/Display an Employee.**

From the Plans table, we see that the plan with ID 2 has userType ‘employee’

planID	name	price	applyLimit	postLimit	userType
1	Employee Basic	0	0	0	employee
2	Employee Prime	10	5	0	employee
3	Employee Gold	20	NULL	0	employee
4	Employer Prime	50	0	5	employer
5	Employer Gold	100	0	NULL	employer
6	Admin	0	0	0	admin
NULL	NULL	NULL	NULL	NULL	NULL

CREATE

```
INSERT INTO Users (userID,
                   planID,
                   email,
                   password,
                   isActive,
                   balance,
                   isAutomatic)
VALUES            ('gordon',
                    2,
                    'gordon@comp353.com',
                    'gordon',
                    TRUE,
                    0,
                    TRUE)
```

userNumber	userID	planID	email	password	dateCreated	isActive	startSufferingDate	balance	isAutomatic
1	gordon	2	gordon@comp353.com	gordon	2020-01-14 23:00:00	1	NULL	0	1

EDIT

```
UPDATE Users
SET isActive = 0
WHERE userNumber = 1;
```

DISPLAY

```
SELECT * FROM Users
WHERE userNumber = 1;
```

DELETE

```
DELETE FROM Users
WHERE userNumber = 1;
```

Note: This delete will cascade and delete rows from all the other tables where the userID ‘gordon’ is a FK.

8. **Search for a job by an employee**

```
SELECT title, datePosted, description, companyName, city, salary, positionsAvailable, status
  FROM Jobs
  JOIN Location ON Jobs.locationID = Location.locationID
  JOIN Profiles ON Jobs.userID = Profiles.userID
  JOIN Users ON Users.userID = Jobs.userID
  JOIN Employer_Categories EC ON Jobs.userID = EC.userID
  WHERE LOWER(title) LIKE LOWER('%Developer%')
  OR LOWER(description) LIKE LOWER('%Developer%')
  OR LOWER(Profiles.companyName) LIKE LOWER('%Developer%')
  OR LOWER(Profiles.firstName) LIKE LOWER('%Developer%')
  OR LOWER(Profiles.lastName) LIKE LOWER('%Developer%')
  OR LOWER(Jobs.userID) LIKE LOWER('%Developer%')
  OR LOWER(Location.city) LIKE LOWER('%Developer%');
```

title	datePosted	description	companyName	city	salary	positionsAvailable	status
Software Developer	2020-07-15 00:00:00	Must know C++	BigMusic	Burlington	90000	3	active
Python Developer	2020-08-01 00:00:00	Must know snakes	Logic Industries	North Pender Island	80000	5	active
Java Developer	2020-08-02 00:00:00	Must like coffee	Logic Industries	Cobble Hill	60000	2	active
JavaScript Developer	2020-08-03 00:00:00	Must know all frameworks	Logic Industries	Pickering	70000	8	active
Rust Developer	2020-08-04 00:00:00	We will ask about metals	Logic Industries	Ile Perrot	85000	3	active
Software Developer	2020-08-06 21:40:38	Work on Angular apps	Logic Industries	Montreal	100000	4	active
Software Developer	2020-07-15 00:00:00	Must have 15 years of experience in PHP	Siens	Brossard	85000	3	active

9. Apply for a job by an employee.

Existing jobs in the Jobs table:

jobID	userID	locationID	title	salary	description	positionsAvailable	datePosted	status
1	leo	1	Software Developer	85000	Must have 15 years of experience in PHP	3	2020-07-15 00:00:00	active
2	bob	2	Human Resources	120000	Must have 20 years experience at any company	1	2020-08-02 00:00:00	active
3	sujan	3	IT Help Desk	60000	Required Skills: Java, mySQL	1	2020-08-13 00:00:00	expired
4	jcole	4	Business Analyst	85000	Requirements: Bachelors in Business	5	2020-07-15 00:00:00	expired
5	jcole	4	Software Developer	90000	Must know C++	3	2020-07-15 00:00:00	active
6	ariana	5	Database Administrator	60000	Looking for McGill students only	1	2020-07-06 00:00:00	active
7	joe	6	Python Developer	80000	Must know snakes	5	2020-08-01 00:00:00	active
8	joe	7	Java Developer	60000	Must like coffee	2	2020-08-02 00:00:00	active
9	joe	8	JavaScript Developer	70000	Must know all frameworks	8	2020-08-03 00:00:00	active
10	joe	9	Rust Developer	85000	We will ask about metals	3	2020-08-04 00:00:00	active
12	joe	11	Software Developer	100000	Work on Angular apps	4	2020-08-06 21:40:38	active
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query:
INSERT INTO Applications (jobID, userID) VALUES (2, ‘tyson’)

New Row in Applications table:

jobID	userID	dateApplied	isAcceptedByEmployer	isAcceptedByEmployee
2	tyson	2020-08-01 17:51:19	NULL	NULL

10. Accept/Deny a job offer by an employee

Existing applications in the Applications table

jobID	userID	dateApplied	isAcceptedByEmployer	isAcceptedByEmployee
3	leo	2020-08-01 19:18:08	1	NULL
2	tyson	2020-08-01 17:51:19	1	NULL

Deny job offer:

```
UPDATE Applications
SET isAcceptedByEmployee = 0
WHERE jobID = 2 AND userID = 'tyson';
```

Accept job offer:

```
UPDATE Applications
SET isAcceptedByEmployee = 1
WHERE jobID = 3 AND userID = 'leo';
```

Applications after user ‘leo’ accepted and user ‘tyson’ denies

jobID	userID	dateApplied	isAcceptedByEmployer	isAcceptedByEmployee
3	leo	2020-08-01 19:18:08	1	1
2	tyson	2020-08-01 17:51:19	1	0

11. Withdraw from an applied job by an employee.

```
DELETE FROM Applications
WHERE jobID = 1 and userID = 'leo';
```

12. Delete a profile by an employee.

```
DELETE FROM Profiles
WHERE userID = 'gordon';
```

13. **Report of applied jobs by an employee during a specific period of time (Job title, date applied, short description of the job up to 50 characters, status of the application).**

```
SELECT j.title, SUBSTRING(j.description, 1, 50) AS 'description', a.dateApplied,
a.isAcceptedByEmployer, a.isAcceptedByEmployee
FROM Applications AS a, Jobs as j
WHERE a.userID = [input.userID] AND j.jobID = a.jobID
AND a.dateApplied BETWEEN 'YYYY-MM-DD' AND 'YYYY-MM-DD'
GROUP BY j.jobID;
```

title	description	dateApplied	isAcceptedByEmployer	isAcceptedByEmployee
Software Developer	Must have 15 years of experience in PHP	2020-08-06 15:54:37	1	NULL
Human Resources	Must have 20 years experience at any company	2020-08-06 15:54:37	NULL	NULL
IT Help Desk	Required Skills: Java, mySQL	2020-08-06 15:54:37	NULL	NULL
Business Analyst	Requirements: Bachelors in Business	2020-08-06 15:54:37	NULL	NULL
Software Developer	Must know C++	2020-08-06 15:54:37	NULL	NULL

14. **Add/Delete/Edit a method of payment by a user.**

```
INSERT INTO Payment_Methods (
            userID,
            isPreSelected,
            cardNumber,
            paymentType)
VALUES      ('tyson',
             0,
             1111111111,
              'credit')
```

paymentMethodID	userID	isPreSelected	paymentType	cardNumber
14	tyson	0	credit	123515
NULL	NULL	NULL	NULL	NULL

DELETE

```
DELETE FROM Payment_Methods
WHERE userID = 'tyson' AND paymentMethodID = 14
```

EDIT

```
UPDATE Payment_Methods
SET isPreSelected = 1
WHERE userID = 'tyson' AND paymentMethodID = 14
```

15. Add/Delete/Edit an automatic payment by a user.

From the Users table, we see that this user uses automatic payments

userID	planID	email	password	dateCreated	isActive	startSufferingDate	balance	isAutomatic
gordon	1	gordon@comp353.com	gordon	2020-01-15 00:00:00	1	NULL	0	1

Since the user wants to use automatic payments, they will be required to add at least one payment method where isPreSelected = 1. This pre-selected payment method will be the one used by the system to charge the user on the first of the month.

CREATE

```
INSERT INTO Payment_Methods (paymentMethodID,
                userID,
                isPreSelected,
                cardNumber,
                paymentType)
VALUES     (1,
                'gordon',
                1,
                1234,
                debit)
```

paymentMethodID	userID	isPreSelected	paymentType	cardNumber
1	gordon	1	debit	1234

DELETE

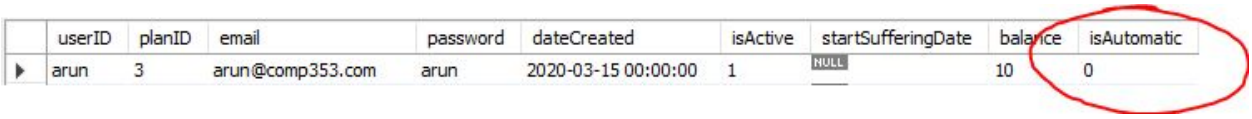
```
DELETE FROM Payment_Methods
WHERE userID = 'gordon' AND paymentMethodID = 1
```

EDIT

```
UPDATE Payment_Methods
SET isPreSelected = 0
WHERE userID = 'gordon' AND paymentMethodID = 1
```

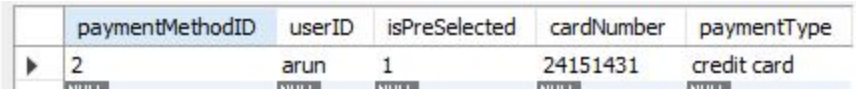
16. Make a manual payment by a user.

From the Users table, we see that this user does not use automatic payments



	userID	planID	email	password	dateCreated	isActive	startSufferingDate	balance	isAutomatic
▶	arun	3	arun@comp353.com	arun	2020-03-15 00:00:00	1	NULL	10	0

In the Payment_Methods table, user ‘arun’ has the following payment methods:

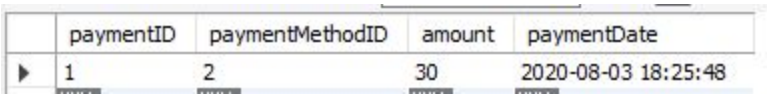


	paymentMethodID	userID	isPreSelected	cardNumber	paymentType
▶	2	arun	1	24151431	credit card

Now when making a payment, we use the following query:

INSERT INTO Payments (paymentMethodID, amount)
VALUES (2, 30);

Content in the payments table:



	paymentID	paymentMethodID	amount	paymentDate
▶	1	2	30	2020-08-03 18:25:48

In the PHP code we then calculate the new balance and perform the following to update the balance.

UPDATE Users SET balance = :newBalance WHERE userID = :user"

17. **Report of all users by the administrator for employers or employees (Name, email, category, status, balance.**

```
SELECT p.firstName as 'First Name', p.lastName as 'Last Name', u.email as 'Email', pl.name
as 'Plan Name', pl.userType as 'User Type', u.isActive, u.balance as 'Balance'
FROM Users as u, Profiles as p, Plans as pl
WHERE u.planID = pl.planID AND u.userID = p.userID
AND pl.userType <> 'admin'
```

First Name	Last Name	Email	Plan Name	User Type	isActive	Balance
gordon	ramsay	gordon@comp353.com	Employee Basic	employee	1	0
tiffany	zeng	tiffany@comp353.com	Employee Prime	employee	1	10
tyson	chandler	tyson@comp353.com	Employee Prime	employee	1	-10
leo	silao	leo@comp353.com	Employer Prime	employer	0	0
bob	thebuilder	bob@comp353.com	Employer Gold	employer	1	-100

18. **Report of all outstanding balance accounts (User name, email, balance, since when the account is suffering).**

```
SELECT userID, email, balance, startSufferingDate
FROM Users
WHERE balance < 0;
```

userID	email	balance	startSufferingDate
leo	leo@comp353.com	-50	2020-08-06 22:40:38
khaled	khaled@comp353.com	-50	2019-06-15 00:00:00
sujan	sujan@comp353.com	-50	2020-08-06 22:42:54
jcole	jcole@comp353.com	-50	2020-08-06 22:42:55
tyson	tyson@comp353.com	-10	2020-03-15 00:00:00