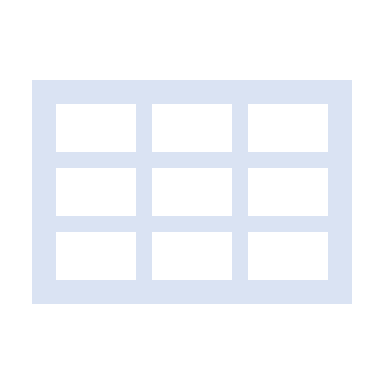
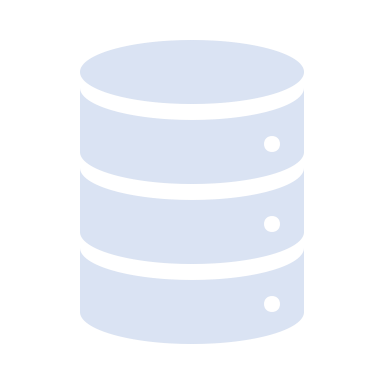
COMP 353 CC

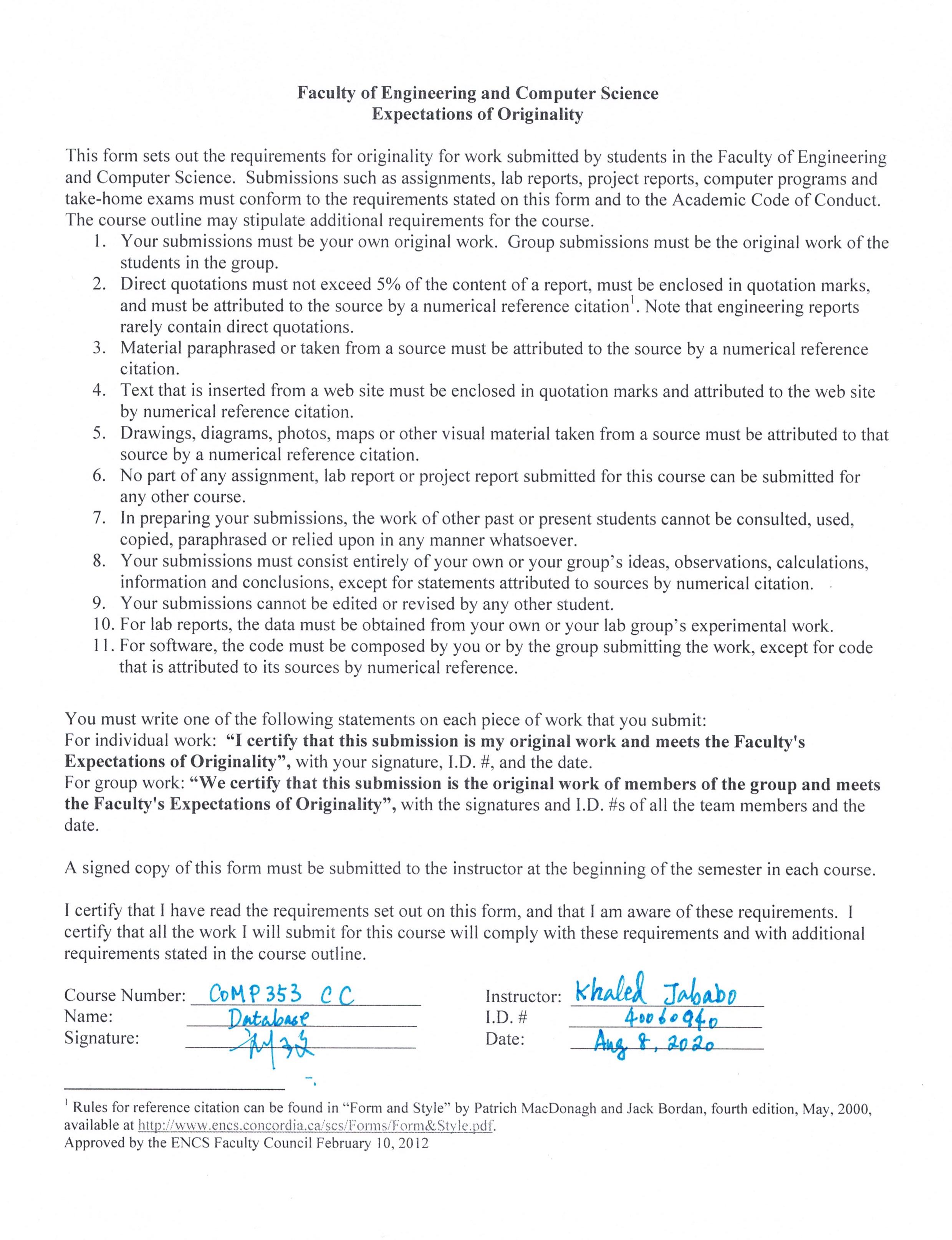
Main Project

Group Account: tyc353\_1



|  |  |
| --- | --- |
| **Member** | **ID** |
| Mei Liu | 40060940 |





**TABLE OF CONTENTS**

[1. Overview 5](#_Toc47897546)

[2. ER Design 5](#_Toc47897547)

[3. Relational database scheme and Normalization 6](#_Toc47897548)

[4. Functionalities. 10](#_Toc47897549)

[5. All Relations Declaration 26](#_Toc47897550)

**LIST OF FIGURES**

Figure 1 : ER Diagram 5

Figure 2: Output of query i of Create 11

Figure 3: Output of query i of Delete 11

Figure 4: Output of query i of Edit 12

Figure 5: Output of query i of Display 12

Figure 6: Output of query ii of Create 13

Figure 7: Output of query ii of Delete 14

Figure 8: Output of query ii of Edit 15

Figure 9: Output of query ii of Display 15

Figure 10: Output of query iii 16

Figure 11: Output of query iv 17

Figure 12: Output of query v 18

Figure 13: Output of query vi 18

Figure 14: Output of query vii of Create 19

Figure 15: Output of query vii of Delete 19

Figure 16: Output of query vii of Edit 20

Figure 17: Output of query vii of Display 20

Figure 18: Output of query viii 20

Figure 19: Output of query ix 21

Figure 20: Output of query x of Accept 22

Figure 21: Output of query x of Deny 22

Figure 22: Output of query xi 23

Figure 23: Output of query xii 24

Figure 24: Output of query xiii 24

Figure 25: Output of query xiv of Add 25

Figure 26: Output of query xiv of Delete 25

Figure 27: Output of query xiv of Edit 26

Figure 28: Output of query xv of Create 26

Figure 29: Output of query xv of Delete 27

Figure 30: Output of query xv of Edit 27

Figure 31: Output of query xvi 28

# Overview

This project is aimed to build up a Web Career Portal for employers and employees to facilitate their day to day tasks and provide a better service for their clients.

Based upon requirements for two major entities in this project: Employers and Employees, the reasonable assumptions we have made is that several relations among Employers, Employees and Website Administrator should be created, and accosicated with payment information and job information, such that an effective database could be built up to support all those basic requirements. Meanwhile, we should normalize all relations in 3NF so as to avoid redundant data as much as possible.

# ER Design

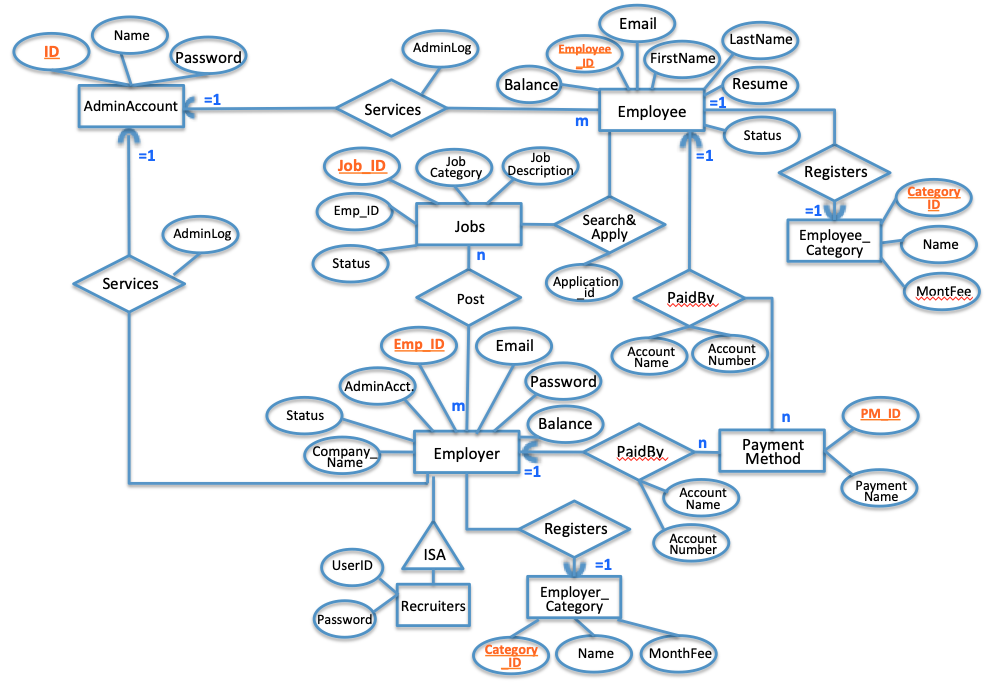
According to the requirement, the figure 1 is the the designed ER diagram, which illustrates the conceptual database scheme.

Figure 1 : ER Diagram

# Relational database scheme and Normalization

Based the ER design, the relational database scheme Identify the primary keys and the foreign keys; and the referential integrity constraints is also noted.

Normalization of 3NF is also indicated by showing FD sets and candidate key identification.

**ADMIN\_ACCOUNT** (**nam**e, password, email)

Primary key: name

Foreign key: nil

NORMALIZATION:

F = {name 🡪 password, name🡪email}

Candidate key = name

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**ADMIN\_LOG** (**log\_id**, log\_time, activity)

Primary key: log\_id

Foreign key: nil

NORMALIZATION:

F = {log\_id 🡪 log\_time, log\_id 🡪 activity }

Candidate key = log\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**EMPLOYER** (**employer\_id,** name, password, company, email, category, status, balance, last\_pay\_time, auto\_pay, default\_pm\_id)

Primary Key: employer\_id

Foreign Key:category, pm\_id

Notes for Referential Integrity:

* The category must match a valid category\_id in the relation EMPLOYER\_CATEGORY
* The default\_pm\_id must match a valid pm\_id in the PAYMENT\_EMPLOYER

Normalization:

F = { employer\_id 🡪 name, employer\_id 🡪 password, employer\_id 🡪 company, employer\_id 🡪 email, employer\_id 🡪 category, employer\_id 🡪 status, employer\_id 🡪 balance, employer\_id 🡪 last\_pay\_time, employer\_id 🡪 auto\_pay, employer\_id 🡪 pm\_id}

Candidate key = employer\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**EMPLOYER\_CATEGORY (category\_id**, category, month\_fee)

Primary Key: category\_id

Foreign Key:nil

Normalization:

F = { category\_id 🡪 category, category\_id 🡪 month\_fee}

Candidate key = category\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**PAYMENT\_EMPLOYER** (**pm\_id**, employer\_id, pay\_method, account\_name, account\_number)

Primary Key: pm\_id

Foreign Key:employer\_id

Notes for Referential Integrity:

* The employer\_id must match a valid employer\_id in the relation EMPLOYER.

Normalization:

F = { pm\_id 🡪 employer\_id, pm\_id 🡪 pay\_method, pm\_id 🡪 account\_name, pm\_id 🡪 account\_number }

Candidate key = employer\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**RECRUITER** (**recruiter\_id**, employer\_id, name, password, first\_name, last\_name, email)

Primary Key: recruiter\_id

Foreign Key:employer\_id

Notes for Referential Integrity:

* The employer\_id must match a valid employer\_id in the relation EMPLOYER.

Normalization:

F = { recruiter\_id 🡪 employer\_id, recruiter\_id 🡪 name, recruiter\_id 🡪 first\_name, recruiter\_id 🡪 last\_name, recruiter\_id 🡪 email }

Candidate key = recruiter\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**EMPLOYEE** (**employee\_id**, name, first\_name, last\_name, email, category, status, balance, resume, last\_pay\_time, auto\_pay, default\_pm\_id)

Primary Key: employee\_id

Foreign Key:default\_pm\_id, category

Notes for Referential Integrity:

* The default\_pm\_id must match a valid pm\_id in the relation PAYMENT\_EMPLOYEE.
* The category must match a valid category id in the relation EMPLOYEECATEGORY.

Normalization:

F = { employee\_id 🡪 name, employee\_id 🡪 first\_name, employee\_id 🡪 last\_name, employee\_id 🡪 email , employee\_id 🡪 category , employee\_id 🡪 status , employee\_id 🡪 balance , employee\_id 🡪 resume , employee\_id 🡪 last\_pay\_time , employee\_id 🡪 auto\_pay, employee\_id 🡪 default\_pm\_id}

Candidate key = employee\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**EMPLOYEE\_CATEGORY (category\_id**, category, month\_fee)

Primary Key: category\_id

Foreign Key:nil

Normalization:

F = { category\_id 🡪 category, category\_id 🡪 month\_fee}

Candidate key = category\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**PAYMENT\_EMPLOYEE** (**pm\_id**, employer\_id, pay\_method, account\_name, account\_number)

Primary Key: pm\_id

Foreign Key:employee\_id

Notes for Referential Integrity:

* The employer\_id must match a valid employer\_id in the relation EMPLOYEE.

Normalization:

F = { pm\_id 🡪 employee\_id, pm\_id 🡪 pay\_method, pm\_id 🡪 account\_name, pm\_id 🡪 account\_number }

Candidate key = employer\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**AUTO\_PAY** (**auto\_pay**, value)

Primary Key: auto\_pay

Foreign Key:nil

Normalization:

F = { auto\_pay 🡪 value}

Candidate key = auto\_pay

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**PAYMENT\_METHOD** (**method\_id**, method\_name)

Primary Key: method\_id

Foreign Key:nil

Normalization:

F = { method\_id 🡪 method\_name}

Candidate key = method\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**JOB** (**job\_id**, employer\_id, title, needed, job\_cid, description, status, post\_time)

Primary Key: job\_id

Foreign Key:employee\_id, job\_cid

Notes for Referential Integrity:

* The employer\_id must match a valid employer\_id in the relation EMPLOYEE.
* The job\_cid must match a valid job\_cid in the relation JOB\_CATEGORY

Normalization:

F = { job\_id 🡪 employee\_id, job\_id 🡪 title, job\_id 🡪 needed, job\_id 🡪 job\_cid, job\_id 🡪 description, job\_id 🡪 status, job\_id 🡪 post\_time}

Candidate key = job\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**JOB\_CATEGORY** (**job\_cid**, employer\_id, category)

Primary Key: job\_cid

Foreign Key:employee\_id

Notes for Referential Integrity:

* The employer\_id must match a valid employer\_id in the relation EMPLOYEE.

Normalization:

F = { job\_cid 🡪 employee\_id, job\_cid 🡪 category}

Candidate key = job\_cid

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**APPLICATION** (**app\_id**, job\_id, employee\_id, status\_id, apply\_time, )

Primary Key: app\_id

Foreign Key:employee\_id

Notes for Referential Integrity:

* The employer\_id must match a valid employer\_id in the relation EMPLOYEE.

Normalization:

F = { app\_id 🡪 employee\_id, app\_id 🡪 job\_id, app\_id 🡪 status\_id, app\_id 🡪 apploy\_time}

Candidate key = app\_id

**All LHS of FDs are candidate key, therefore R is in 3NF.**

**APPLICATION\_STATUS** (**status\_id**, status )

Primary Key: status\_id

Foreign Key:nil

Normalization:

F = { status\_id 🡪status}

Candidate key = status

**All LHS of FDs are candidate key, therefore R is in 3NF.**

# Functionalities.

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

Create:

INSERT INTO TB\_EMPLOYER VALUES (108,'admin',1,'Infinity Tech', 'hiring@ift.ca',2,0,0,NULL,1,NULL);

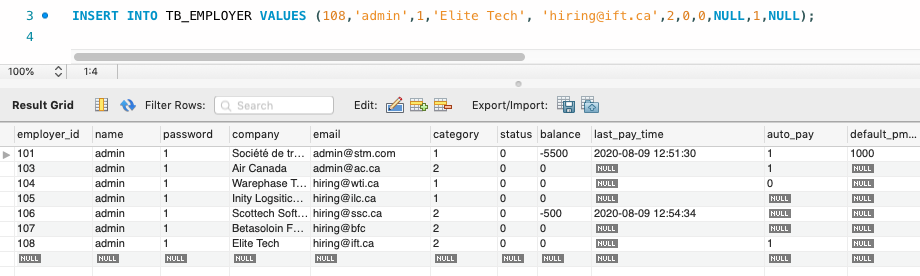


Figure 2: Output of query i of Create

Delete:

DELETE FROM TB\_EMPLOYER WHERE employer\_id=108;

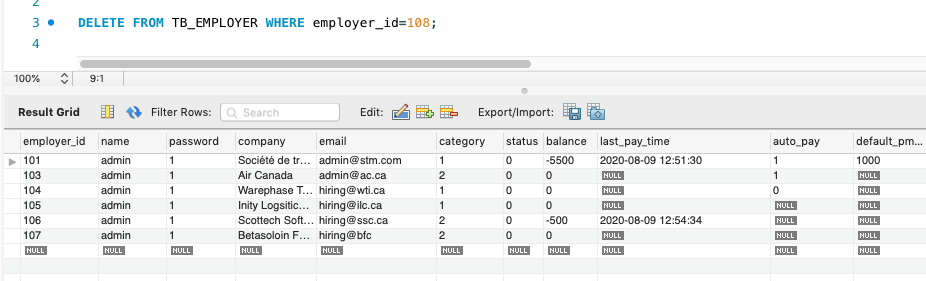


Figure 3: Output of query i of Delete

Edit:

UPDATE TB\_EMPLOYER

SET auto\_pay = 1, category =2

WHERE employer\_id = 103;

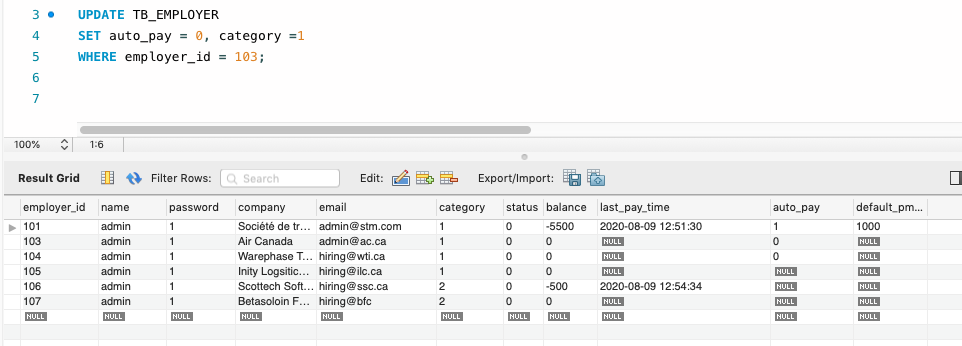


Figure 4: Output of query i of Edit

Display

select \* FROM TB\_EMPLOYER

WHERE employer\_id = 103;

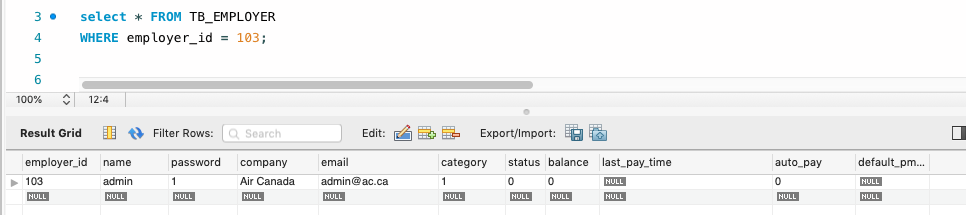


Figure 5: Output of query i of Display

ii Create/Delete/Edit/Display a category by an Employer.

Create:

INSERT INTO TB\_JOB\_CATEGORY (employer\_id,category) VALUES (105,'Engineering');

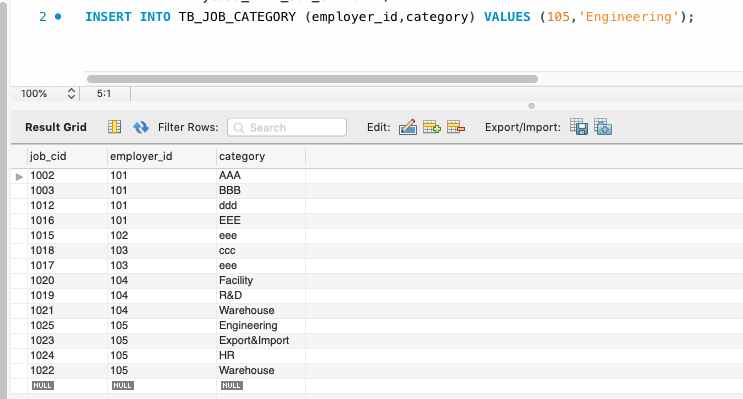


Figure 6: Output of query ii of Create

Delete:

DELETE FROM TB\_JOB\_CATEGORY WHERE job\_cid = 1025;

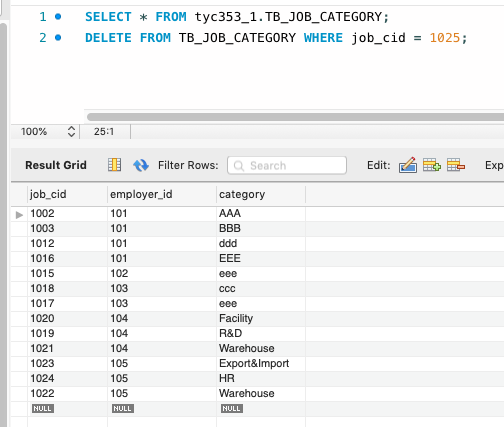


Figure 7: Output of query ii of Delete

Edit:

UPDATE TB\_JOB\_CATEGORY

SET category = ‘QualityControl’

WHERE job\_cid = 1026;

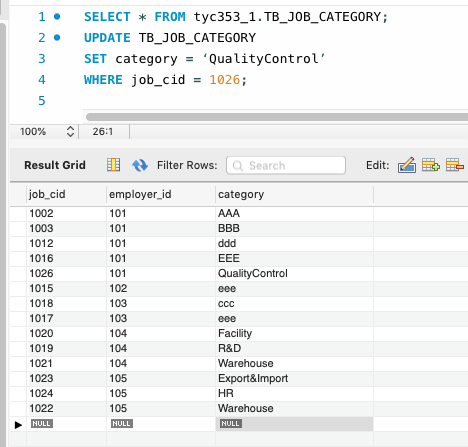


Figure 8: Output of query ii of Edit

Display

select \* FROM TB\_JOB\_CATEGORY

WHERE job\_cid = 1002;

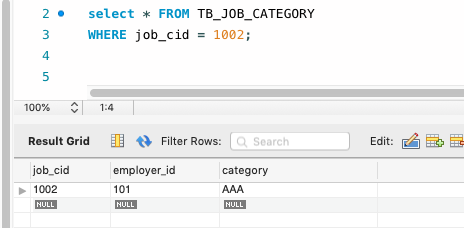


Figure 9: Output of query ii of Display

iii.Post a new job by an employer.

INSERT INTO TB\_JOB VALUES (100022,'105','Tester', 1, 1022, 'Bachelor degree', 1,now());

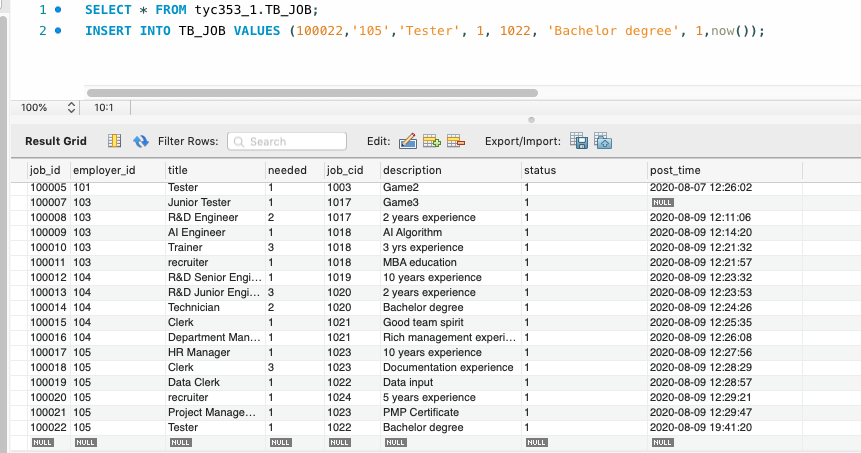


Figure 10: Output of query iii

iv.Provide a job offer for an employee by an employer.

UPDATE TB\_APPLICATION

SET status\_id=1

WHERE app\_id = 1000007;

Remarks: status = 1 means the job is offered to employee\_id = ‘100004’

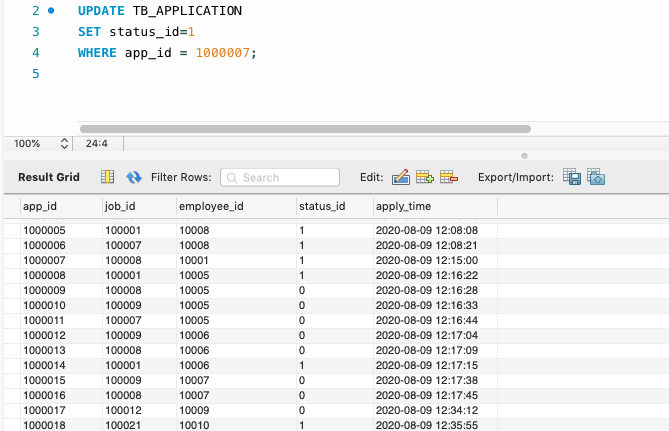


Figure 11: Output of query iv

v. 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 app\_id,a.job\_id,title,company,apply\_time,a.status\_id,s.status, ee.first\_name as 'applicant first name', ee.last\_name as 'applicant last name'

FROM TB\_APPLICATION as a, TB\_JOB as j, TB\_EMPLOYER as e, TB\_APPLICATION\_STATUS as s, TB\_EMPLOYEE as ee

WHERE e.employer\_id=101 AND

j.job\_id = 100001 AND

a.job\_id = j.job\_id AND

j.employer\_id = e.employer\_id AND

a.status\_id = s.status\_id AND

a.employee\_id = ee.employee\_id;

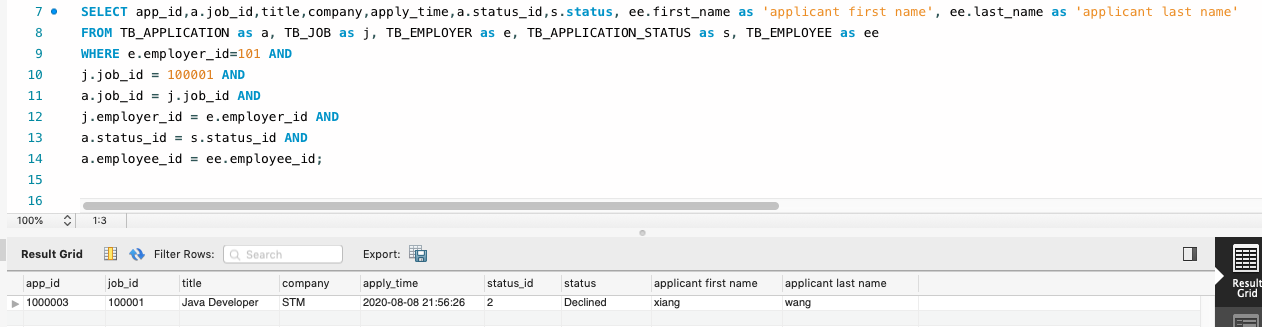


Figure 12: Output of query v

vi. 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, j.post\_time, j.description, j.needed, COUNT(a.employee\_id), COUNT(s.status = 'Accepted'OR NULL)

FROM TB\_JOB AS j, TB\_APPLICATION AS a, TB\_APPLICATION\_STATUS as s

where j.job\_id=a.job\_id and

a.status\_id = s.status\_id and

# s.status\_id = 1 and

j.employer\_id = 101 and

j.post\_time > '2020-08-01' and

j.post\_time < '2020-09-01'

GROUP BY j.job\_id;

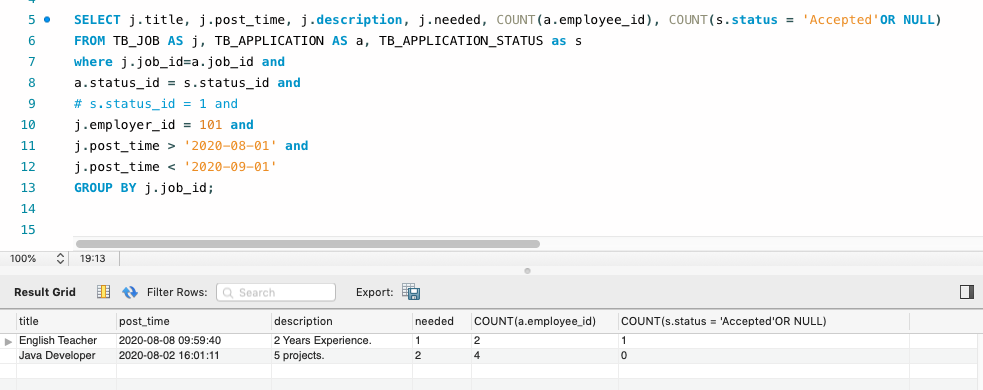


Figure 13: Output of query vi

vii. Create/Delete/Edit/Display an Employee.

Create:

INSERT INTO TB\_EMPLOYEE VALUES ('10012', 'fayer', '1', 'fayer', 'farmer', 'ffarmer@yahoo.com', '0', '0', '2600', 'resumefayer', NULL, '1', NULL);

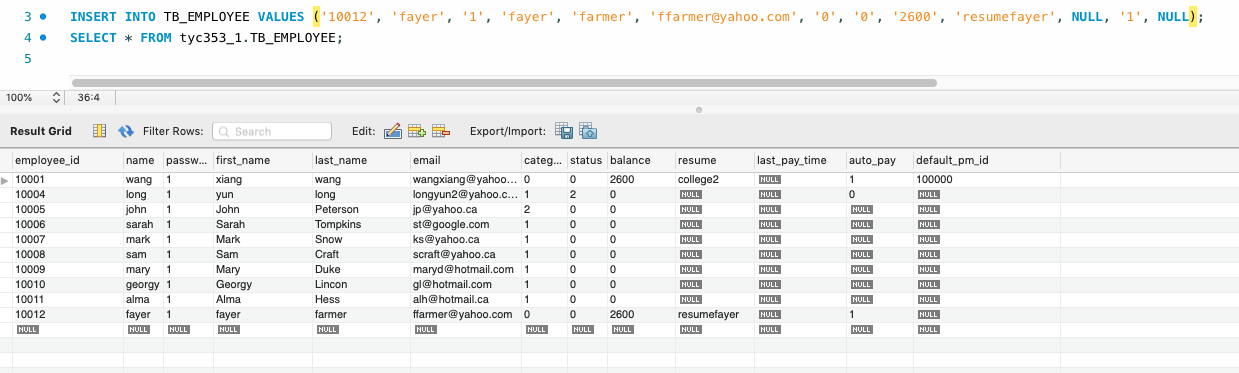


Figure 14: Output of query vii of Create

Delete:

DELETE FROM TB\_EMPLOYEE WHERE employee\_id=10012;

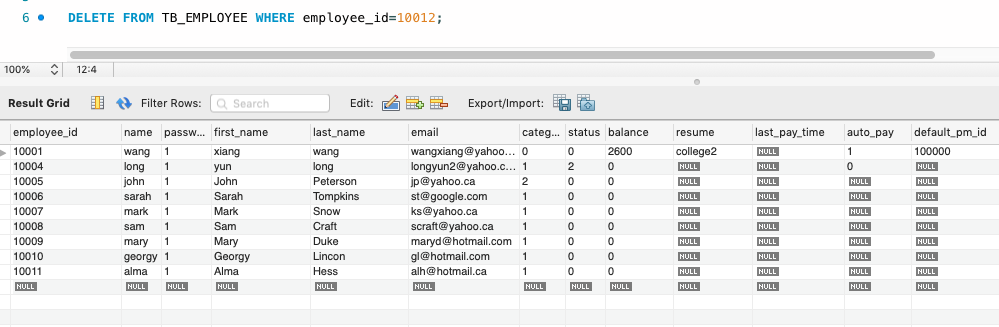


Figure 15: Output of query vii of Delete

Edit:

UPDATE TB\_EMPLOYEE

SET auto\_pay = 1, category =2

WHERE employee\_id = 10010;

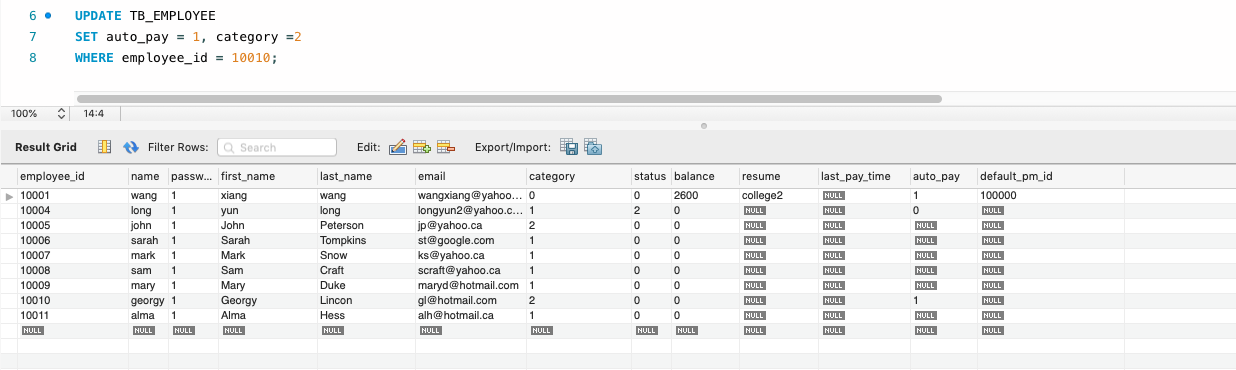


Figure 16: Output of query vii of Edit

Display:

select \* FROM TB\_EMPLOYEE

WHERE employee\_id = 10011;

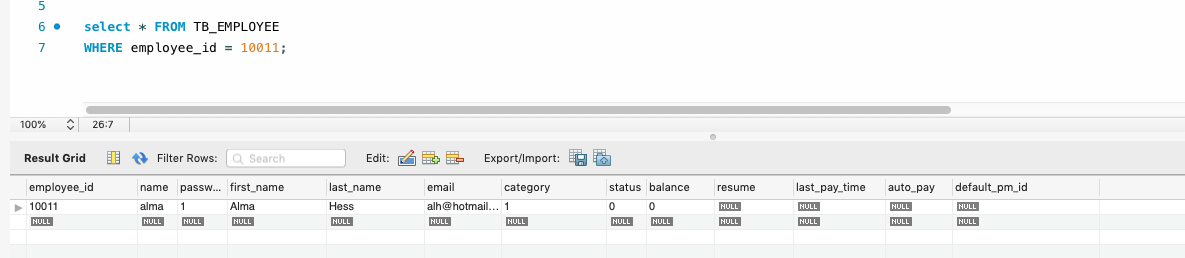


Figure 17: Output of query vii of Display

viii. Search for a job by an employee.

select \*

FROM TB\_JOB AS J

where J.title ='R&D Engineer';

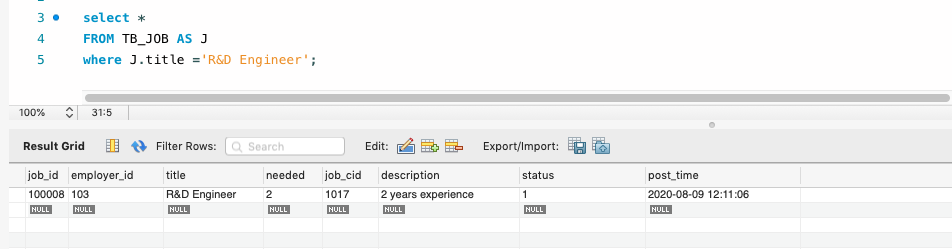


Figure 18: Output of query viii

ix. Apply for a job by an employee.

INSERT INTO TB\_APPLICATION (job\_id,employee\_id,status\_id,apply\_time) VALUES (100007,10011,0,now());

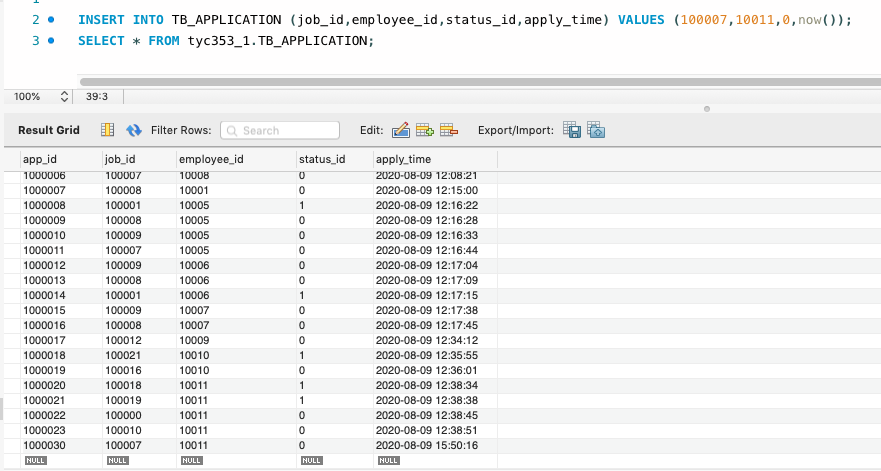


Figure 19: Output of query ix

x. Accept/Deny a job offer by an employee.

Accept:

UPDATE TB\_APPLICATION

SET status\_id = 3

WHERE employee\_id = 10011 and

job\_id = 100019;

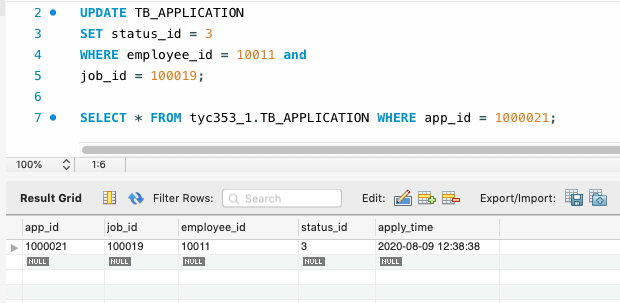


Figure 20: Output of query x of Accept

Deny:

UPDATE TB\_APPLICATION

SET status\_id = 2

WHERE employee\_id = 10011 and

job\_id = 100019;

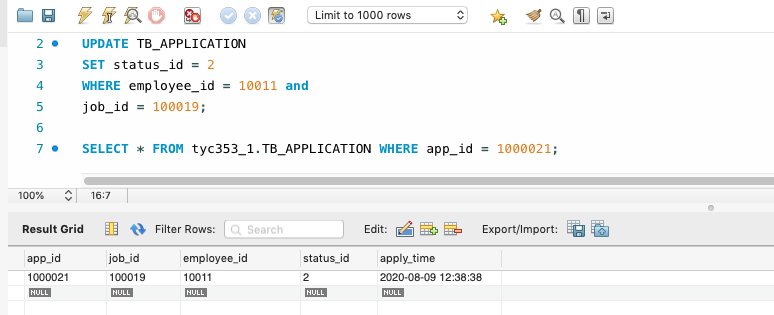


Figure 21: Output of query x of Deny

xi. Withdraw from an applied job by an employee.

DELETE FROM TB\_APPLICATION

WHERE employee\_id=1006 and

app\_id=1000013;

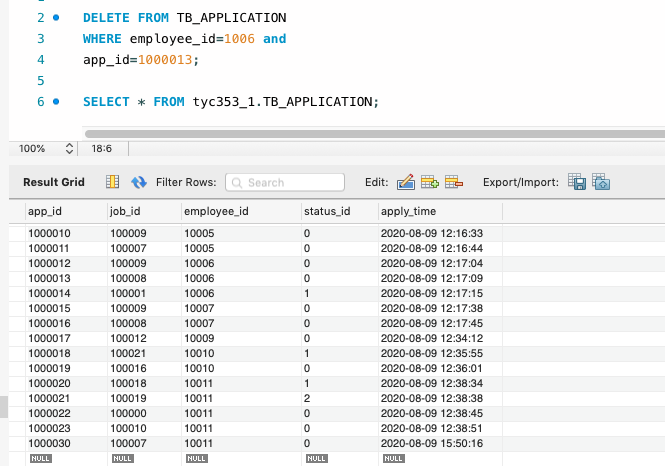


Figure 22: Output of query xi

xii. Delete a profile by an employee.

UPDATE TB\_EMPLOYEE

SET resume = NULL

WHERE employee\_id = 10004;

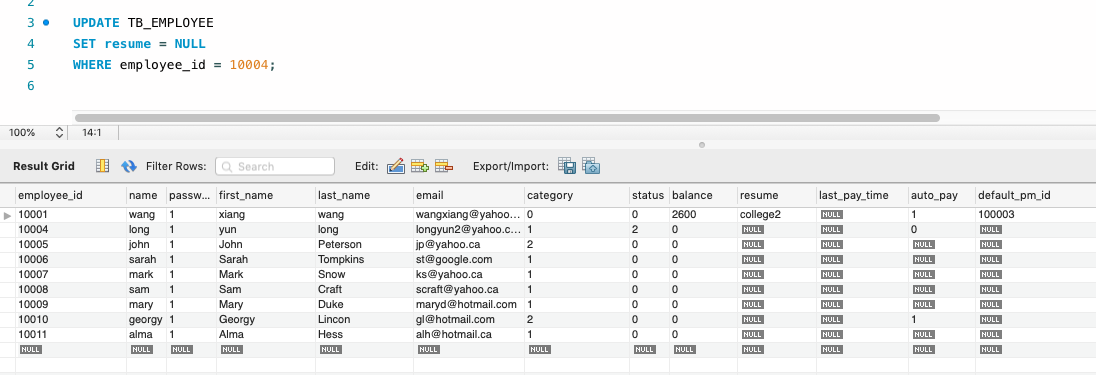


Figure 23: Output of query xii

xiii. 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, a.apply\_time, j.description, s.status

FROM TB\_JOB AS j, TB\_APPLICATION AS a, TB\_APPLICATION\_STATUS as s

where j.job\_id=a.job\_id and

a.status\_id = s.status\_id and

a.employee\_id = 10001;

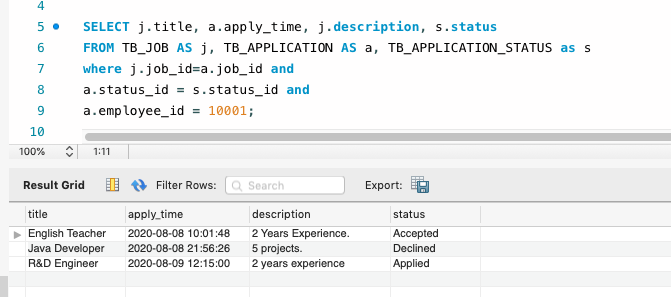


Figure 24: Output of query xiii

xiv. Add/Delete/Edit a method of payment by a user.

Add:

INSERT INTO TB\_PAYMENT\_EMPLOYER VALUES (1002,101, 1, 'Bob Smith', 100-323-333);

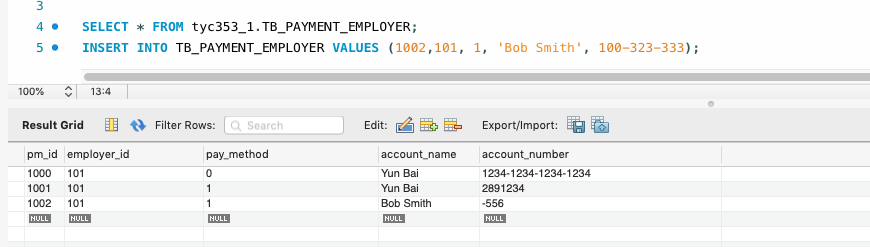


Figure 25: Output of query xiv of Add

Delete:

DELETE FROM TB\_PAYMENT\_EMPLOYEE WHERE pm\_id=1002;

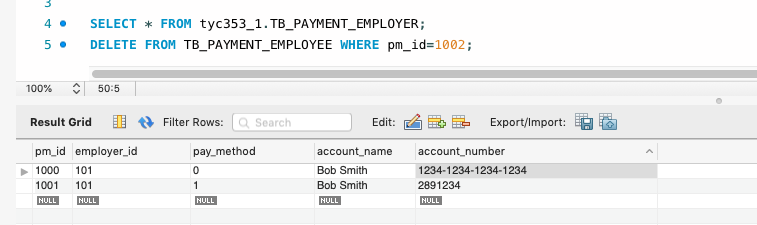


Figure 26: Output of query xiv of Delete

Edit:

UPDATE TB\_PAYMENT\_EMPLOYEE

SET pay\_method = 0

WHERE pm\_id = 1001;

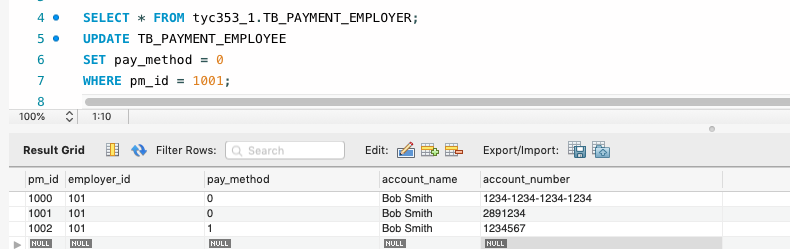


Figure 27: Output of query xiv of Edit

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

Create:

UPDATE TB\_EMPLOYER

SET auto\_pay = 0

WHERE employer\_id = 104;

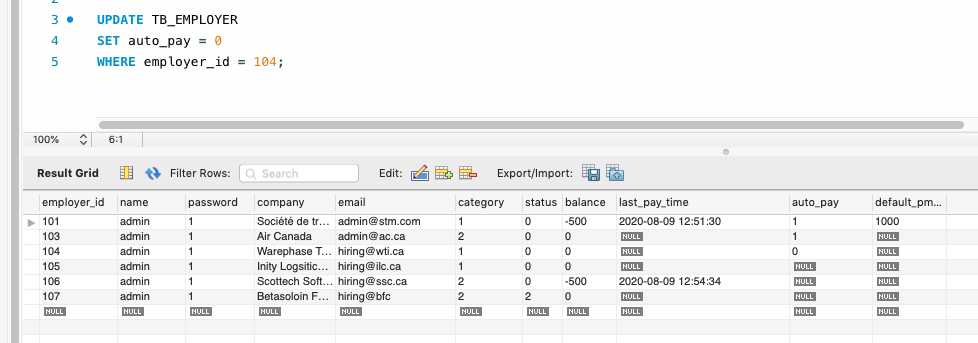


Figure 28: Output of query xv of Create

Delete:

UPDATE TB\_EMPLOYER

SET auto\_pay = null

WHERE employer\_id = 104;

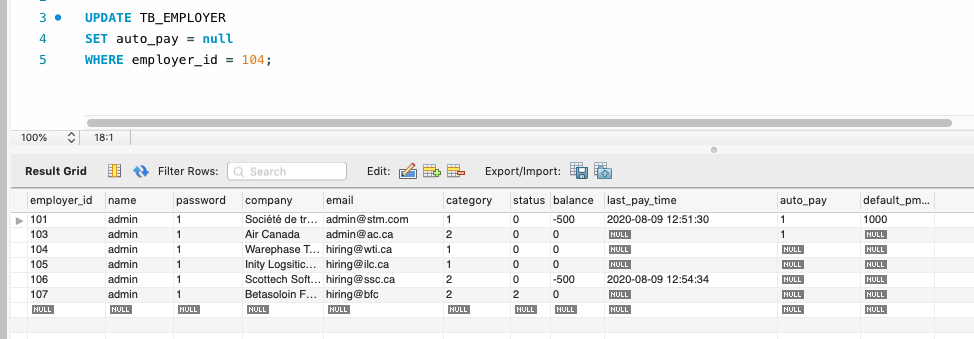


Figure 29: Output of query xv of Delete

Edit:

UPDATE TB\_EMPLOYER

SET auto\_pay = 0

WHERE employer\_id = 104;

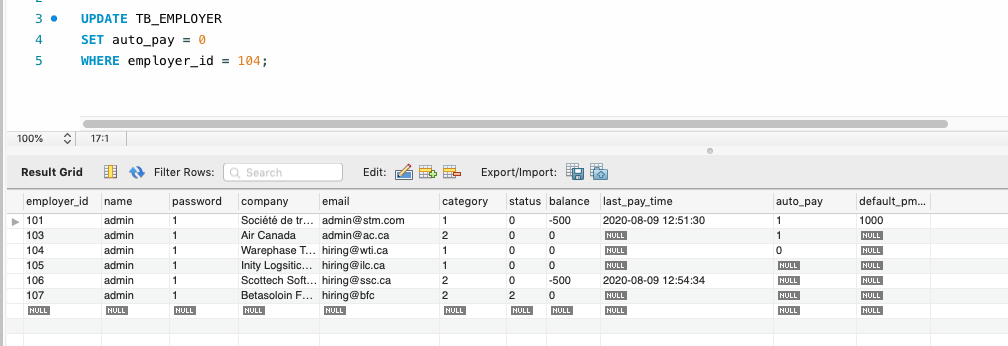


Figure 30: Output of query xv of Edit

xvi. Make a manual payment by a user.

UPDATE TB\_EMPLOYEE

SET balance = balance - 200

WHERE employee\_id = 10001;

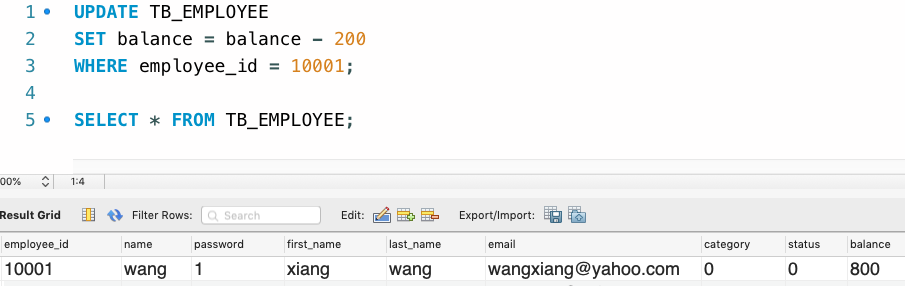


Figure 31: Output of query xvi

# All Relations Declaration

CREATE TABLE TB\_ADMIN\_ACCOUNT (

name varchar(45) NOT NULL,

password varchar(45) NOT NULL,

email varchar(45) NOT NULL,

PRIMARY KEY (name)

) ;

CREATE TABLE TB\_ADMIN\_LOG (

log\_id int NOT NULL AUTO\_INCREMENT,

log\_time datetime DEFAULT NULL,

activity varchar(300) DEFAULT NULL,

PRIMARY KEY (log\_id)

);

CREATE TABLE TB\_APPLICATION\_STATUS (

status\_id int NOT NULL,

status varchar(45) NOT NULL,

PRIMARY KEY (status\_id),

UNIQUE KEY status\_UNIQUE (status)

);

CREATE TABLE TB\_EMPLOYEE\_CATEGORY (

category\_id int NOT NULL,

category varchar(45) NOT NULL,

month\_fee int NOT NULL,

limit int NOT NULL,

PRIMARY KEY (category\_id),

UNIQUE KEY category\_UNIQUE (category)

);

CREATE TABLE TB\_EMPLOYER\_CATEGORY (

category\_id int NOT NULL,

category varchar(45) NOT NULL,

month\_fee int NOT NULL,

limit int NOT NULL,

PRIMARY KEY (category\_id),

UNIQUE KEY category\_UNIQUE (category)

);

CREATE TABLE TB\_AUTO\_PAY (

auto\_pay int NOT NULL,

value varchar(45) NOT NULL,

PRIMARY KEY (auto\_pay),

UNIQUE KEY value\_UNIQUE (value)

);

CREATE TABLE TB\_PAYMENT\_METHOD (

method\_id int NOT NULL,

method\_name varchar(45) NOT NULL,

PRIMARY KEY (method\_id),

UNIQUE KEY method\_name\_UNIQUE (method\_name)

);

CREATE TABLE TB\_EMPLOYEE (

employee\_id int NOT NULL AUTO\_INCREMENT,

name varchar(45) COLLATE utf8\_unicode\_ci NOT NULL,

password varchar(45) COLLATE utf8\_unicode\_ci NOT NULL,

first\_name varchar(45) COLLATE utf8\_unicode\_ci DEFAULT NULL,

last\_name varchar(45) COLLATE utf8\_unicode\_ci DEFAULT NULL,

email varchar(45) COLLATE utf8\_unicode\_ci NOT NULL,

category int DEFAULT NULL,

status int DEFAULT NULL,

balance int( DEFAULT NULL,

resume varchar(1000) COLLATE utf8\_unicode\_ci DEFAULT NULL,

last\_pay\_time datetime DEFAULT NULL,

auto\_pay int DEFAULT NULL,

default\_pm\_id int DEFAULT NULL,

PRIMARY KEY (employee\_id),

UNIQUE KEY name\_UNIQUE (name),

KEY employee\_category\_idx (category),

KEY employee\_auto\_pay\_idx (auto\_pay),

CONSTRAINT employee\_auto\_pay FOREIGN KEY (auto\_pay) REFERENCES TB\_AUTO\_PAY (auto\_pay) ON DELETE RESTRICT,

CONSTRAINT employee\_category FOREIGN KEY (category) REFERENCES TB\_EMPLOYEE\_CATEGORY (category\_id)

) AUTO\_INCREMENT=10000;

CREATE TABLE TB\_EMPLOYER (

employer\_id int NOT NULL AUTO\_INCREMENT,

name varchar(45) COLLATE utf8\_unicode\_ci NOT NULL,

password varchar(45) COLLATE utf8\_unicode\_ci DEFAULT NULL,

company varchar(45) COLLATE utf8\_unicode\_ci DEFAULT NULL,

email varchar(45) COLLATE utf8\_unicode\_ci DEFAULT NULL,

category int DEFAULT NULL,

status int DEFAULT NULL,

balance int DEFAULT NULL,

last\_pay\_time datetime DEFAULT NULL,

auto\_pay int DEFAULT NULL,

default\_pm\_id int DEFAULT NULL,

PRIMARY KEY (employer\_id),

KEY employer\_catagory\_idx (category),

KEY employer\_auto\_pay\_idx (auto\_pay),

CONSTRAINT employer\_auto\_pay FOREIGN KEY (auto\_pay) REFERENCES TB\_AUTO\_PAY (auto\_pay),

CONSTRAINT employer\_catagory FOREIGN KEY (category) REFERENCES TB\_EMPLOYER\_CATEGORY (category\_id)

) AUTO\_INCREMENT=100;

CREATE TABLE TB\_JOB\_CATEGORY (

job\_cid int NOT NULL AUTO\_INCREMENT,

employer\_id int NOT NULL,

category varchar(45) NOT NULL,

PRIMARY KEY (job\_cid),

KEY job\_category (employer\_id,category)

);

CREATE TABLE TB\_JOB (

job\_id int NOT NULL AUTO\_INCREMENT,

employer\_id int NOT NULL,

title varchar(45) NOT NULL,

needed int NOT NULL,

job\_cid int NOT NULL,

description varchar(50) DEFAULT NULL,

status int DEFAULT NULL,

post\_time datetime DEFAULT NULL,

PRIMARY KEY (job\_id),

KEY job\_cid\_idx (job\_cid),

KEY employer\_id\_idx (employer\_id),

CONSTRAINT job\_cid FOREIGN KEY (job\_cid) REFERENCES TB\_JOB\_CATEGORY (job\_cid) ON DELETE RESTRICT,

CONSTRAINT job\_employer FOREIGN KEY (employer\_id) REFERENCES TB\_EMPLOYER (employer\_id) ON DELETE RESTRICT

) AUTO\_INCREMENT=100000;

CREATE TABLE TB\_PAYMENT\_EMPLOYEE (

pm\_id int NOT NULL AUTO\_INCREMENT,

employee\_id int NOT NULL,

pay\_method int DEFAULT NULL,

account\_name varchar(45) DEFAULT NULL,

account\_number varchar(45) DEFAULT NULL,

PRIMARY KEY (pm\_id),

KEY pm\_employee\_idx (employee\_id),

CONSTRAINT pm\_employee FOREIGN KEY (employee\_id) REFERENCES TB\_EMPLOYEE (employee\_id) ON DELETE CASCADE

) AUTO\_INCREMENT=100000;

CREATE TABLE TB\_PAYMENT\_EMPLOYER (

pm\_id int NOT NULL AUTO\_INCREMENT,

employer\_id int NOT NULL,

pay\_method int DEFAULT NULL,

account\_name varchar(45) DEFAULT NULL,

account\_number varchar(45) DEFAULT NULL,

PRIMARY KEY (pm\_id),

KEY pm\_employer\_idx (employer\_id),

CONSTRAINT pm\_employer FOREIGN KEY (employer\_id) REFERENCES TB\_EMPLOYER (employer\_id)

) AUTO\_INCREMENT=1000;

CREATE TABLE TB\_RECRUITER (

recruiter\_id int NOT NULL AUTO\_INCREMENT,

employer\_id int NOT NULL,

name varchar(45) NOT NULL,

password varchar(45) DEFAULT NULL,

first\_name varchar(45) DEFAULT NULL,

last\_name varchar(45) DEFAULT NULL,

email varchar(45) DEFAULT NULL,

PRIMARY KEY (recruiter\_id),

KEY recruiter\_employer\_idx (employer\_id),

CONSTRAINT recruiter\_employer FOREIGN KEY (employer\_id) REFERENCES TB\_EMPLOYER (employer\_id) ON DELETE RESTRICT

) AUTO\_INCREMENT=1000;

CREATE TABLE TB\_APPLICATION (

app\_id int NOT NULL AUTO\_INCREMENT,

job\_id int NOT NULL,

employee\_id int NOT NULL,

status\_id int DEFAULT NULL,

apply\_time datetime DEFAULT NULL,

PRIMARY KEY (app\_id),

KEY app\_job\_idx (job\_id),

KEY app\_employee\_idx (employee\_id),

KEY app\_status\_idx (status\_id),

CONSTRAINT app\_employee FOREIGN KEY (employee\_id) REFERENCES TB\_EMPLOYEE (employee\_id) ON DELETE CASCADE,

CONSTRAINT app\_job FOREIGN KEY (job\_id) REFERENCES TB\_JOB (job\_id) ON DELETE RESTRICT,

CONSTRAINT app\_status FOREIGN KEY (status\_id) REFERENCES TB\_APPLICATION\_STATUS (status\_id) ON DELETE RESTRICT

) AUTO\_INCREMENT=1000000;

ALTER TABLE TB\_EMPLOYEE

ADD INDEX employee\_pm\_id\_idx (default\_pm\_id ASC) VISIBLE;

ALTER TABLE TB\_EMPLOYEE

ADD CONSTRAINT employee\_pm\_id

FOREIGN KEY (default\_pm\_id)

REFERENCES TB\_PAYMENT\_EMPLOYEE (pm\_id)

ON DELETE SET NULL

ON UPDATE NO ACTION;

ALTER TABLE TB\_EMPLOYER

ADD INDEX employer\_pm\_id\_idx (default\_pm\_id ASC) VISIBLE;

ALTER TABLE TB\_EMPLOYER

ADD CONSTRAINT employer\_pm\_id

FOREIGN KEY (default\_pm\_id)

REFERENCES TB\_PAYMENT\_EMPLOYER (pm\_id)

ON DELETE SET NULL

ON UPDATE NO ACTION;