

CPSC 304 Project Cover Page

Milestone #: 4

Date: November 25, 2022

Group Number: 42

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Zhengling Jiang	82219353	l4q3n	zjiang21@student.ubc.ca
Junkai Ding	74511775	d7y9d	17254djk@gmail.com
Yingkai Zhao	35356435	d3h5t	yingkai.zhao@outlook.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

Project Description:

The final project is a combination of visa application, processing and info management system based on a relational database and a web GUI. It modeled the processes of visa application, approval and usage, involving data exchange of entities like applicants, applications, family members, embassies, visas, visa officers, border officers, etc.

Functionalities:

this application succeeded in...

- Providing a web-based user-friendly graphical interface
- Presenting appropriate data to proper users in such a way that they chose
- Allowing applicants to create, submit and track visa applications, which a visa officer, assigned by an embassy, will then review and approve
- Allowing a border officer to record the admission of an applicant
- Allowing an administrator to reset the database and run various meaningful queries about the data
- Streamlining the whole process, enforce necessary constraints and maintain data integrity

Tech Stack:

This application used PHP/Oracle for back-end, and HTML, CSS, and JavaScript for front-end programming.

Relational Schemas:

1. FamilyMembers(**aplctID**: integer, name: string, relationship: string)
 PK: aplctID, name
 EK: aplctID references Applicants
 FD: aplctID, name → relationship
2. Applicants(aplctID: integer, passport: string, name: string, gender: string, birthDate: date, country: string)
 PK: aplctID
 CK: passport
 FD: aplctID → passport, name, gender, birthDate, country
 passport → aplctID, name, gender, birthDate, country
3. Applications(applID: integer, **aplctID**: integer, **void**: integer, **eID**: integer, receiveDate: date, status: string)
 PK: applID
 EK: aplctID references Applicants (cannot be null)
 void references VisaOfficers
 eID references Embassies
 FD: applID → aplctID, void, eID, receiveDate, status
4. TouristApps(**applID**: integer, destination: string)
 PK: applID
 EK: applID references Applications
 FD: applID → destination
5. StudentApps(**applID**: integer, school: string)
 PK: applID
 EK: applID references Applications
 FD: applID → school
6. WorkerApps(**applID**: integer, company: string)
 PK: applID
 EK: applID references Applications
 FD: applID → company
7. Embassies(eID: integer, country: string)
 PK: eID
 CK: country
 FD: eID → country
 country → eID

8. VisaOfficers(voID: integer, **eID**: integer, name: string, experience: integer)
PK: voID
FK: eID references Embassies (cannot be null)
FD: voID → eID, name
9. Visas(viD: integer, **applID**: integer, **aplctID**: integer, **voID**: integer, issueDate: date, **type**: string)
PK: viD
FK: applID references Applications (needs to be unique and not null)
aplctID references Applicants (cannot be null)
voID references VisaOfficers (cannot be null)
type references VisaTypes
FD: viD → applID, aplctID, voID, issueDate, type
10. VisaTypes(type: string, length: integer)
PK: type
FD: type → length
11. Admissions(admID: integer, **aplctID**: integer, **boID**: integer, grantDate: date, **entryPoint**: string)
PK: admID
FK: aplctID references Applicants (cannot be null)
boID references BorderOfficers (cannot be null)
entryPoint references AdmissionEntryPoints
FD: admID → aplctID, boID, grantDate, entryPoint
12. AdmissionEntryPoints(entryPoint: string, entryType: string)
PK: entryPoint
FD: entryPoint → entryType
13. BorderOfficers(boID: integer, name: string)
PK: boID
FD: boID → name

Changes:

1. Error correction: type in Visas and entryPion in Admissions are now FKs
2. Error correction: applID in TouristApps, StudentApps and WorkerApps now has ON DELETE CASCADE constraint
3. VisaOfficers now have a new attribute “experience” to allow for more query types

Data in Relations after Initialization SQL:

```
SQL> SELECT * FROM Embassies;
```

EID	COUNTRY
1	Spain
2	China
3	Australia
4	USA
5	Germany
6	UK

6 rows selected.

```
SQL> SELECT * FROM Applicants;
```

APLCTID	PASSPORT	NAME	G	BIRTHDATE	COUNTRY
1	P00000001	John	M	01-FEB-00	Spain
2	P00000002	Xiaohong	X	02-APR-97	China
3	P00000003	Julia	F	16-DEC-65	Australia
4	P00000004	Benjamin	M	22-MAY-10	USA
5	P00000005	Ella	F	30-AUG-02	Germany
6	P00000006	Emma	F	19-OCT-10	Germany
7	P00000007	Lihua	M	19-OCT-93	China
8	P00000008	Edward	X	07-JUN-88	UK
9	P00000009	Iris	F	25-DEC-01	China

9 rows selected.

```
SQL> SELECT * FROM FamilyMembers;
```

APLCTID	NAME	RELATIONSHIP
1	Joe	Father
2	Xiaoli	Mother
3	Peter	Spouse
3	Andy	Son
4	Daisy	Mother

```
SQL> SELECT * FROM VisaTypes;
```

TY	LENGTH
S1	2
S2	5
T1	1
T2	10
W1	3

```
SQL> SELECT * FROM VisaOfficers;
```

VOID	EID	NAME	EXPERIENCE
1	1	Liam	2
2	2	Noah	10
3	3	William	5
4	4	James	7
5	5	Charles	8
6	5	Samantha	11
7	2	Rachel	9
8	1	Joe	3

8 rows selected.

```
SQL> SELECT * FROM AdmissionEntryPoints;
```

ENT	ENTRY
YVR	Air
BLA	Land
PAC	Water
BCA	Air
AKG	Air

```
SQL> SELECT * FROM BorderOfficers;
```

B0ID	NAME
------	------

1	John
2	David
3	Alfred
4	Justin
5	Amy
6	Melissa
7	Kevin
8	Sam

8 rows selected.

```
SQL> SELECT * FROM Applications;
```

APPID	APLCTID	VOID	EID	RECEIVEDA	STATUS
1	1	1	1	19-JUN-17	Approved
2	2	2	2	20-SEP-18	Approved
3	3	3	3	21-DEC-15	Approved
4	4	4	4	22-JUL-18	Approved
5	5	5	5	23-SEP-20	Approved
6	1		1	07-OCT-19	Received
7	2		2	20-JAN-13	Received
8	3		3	11-NOV-19	Received
9	4		4	09-SEP-18	Received
10	5		5	19-JUL-17	Received
11	1		1	30-OCT-20	Received
12	2		2	01-SEP-21	Received
13	3		3	27-DEC-16	Received
14	4		4	18-SEP-19	Received
15	5	5	5	23-SEP-19	Received
16	2	2	2	23-SEP-19	Approved
17	2	2	2	24-SEP-19	Approved
18	2	2	2	25-SEP-19	Approved
19	2	2	2	26-SEP-19	Approved

19 rows selected.

```
SQL> SELECT * FROM TouristApps;
```

APPID	DESTINATION
1	Vancouver
2	Toronto
3	Calgary
4	Kelowna
5	Ottawa
18	Vancouver

6 rows selected.

```
SQL> SELECT * FROM StudentApps;
```

APPID	SCHOOL
6	UBCV
7	UTSG
8	UAlberta
9	UBCO
10	UOttawa
16	UBCO
17	UBCV

7 rows selected.

```
SQL> SELECT * FROM WorkerApps;
```

APPID	COMPANY
11	Amazon
12	Meta
13	Microsoft
14	Apple
15	Walmart
19	Walmart

6 rows selected.

```
SQL> SELECT * FROM Visas;
```

VID	APPID	APLCTID	VOID	ISSUEDATE	TY
1	1	1	1	15-NOV-20	T1
2	2	2	2	03-DEC-20	T1
3	3	2	3	22-OCT-22	T1
4	4	3	4	31-OCT-21	T1
5	5	5	5	18-NOV-20	T2
6	16	2	2	06-NOV-20	S1
7	17	2	2	05-NOV-20	S2
8	18	2	2	14-JAN-21	T2
9	19	2	2	18-MAR-22	W1

9 rows selected.

```
SQL> SELECT * FROM Admissions;
```

ADMID	APLCTID	BOID	GRANTDATE	ENT
1	1	1	22-FEB-22	YVR
2	2	1	21-SEP-22	YVR
3	3	2	31-MAR-22	BLA
4	4	3	04-JUL-22	PAC
5	5	4	17-JAN-22	AKG
6	2	8	17-JAN-22	BLA
7	2	7	31-JAN-22	AKG
8	2	6	17-NOV-22	YVR

8 rows selected.

Queries:

1. INSERT Operation: /aplct/signupOK.php line 44

```
INSERT INTO Applicants(passport, name, gender, birthDate, country)
VALUES (:passport, :name, :gender, :birthDate, :country)
RETURNING aplctID INTO :aplctID
```

GUI:

* required field

* Name:

* Gender:

* Date of Birth:

* Country:

* Passport:

* How many accompanying family members do you have? ☒ 0 ☐ 1 ☐ 2

[Create Account](#)

Before:

```
SQL> SELECT * FROM Applicants;
```

APLCTID	PASSPORT	NAME	G	BIRTHDATE	COUNTRY
1	P00000001	John	M	01-FEB-00	Spain
2	P00000002	Xiaohong	X	02-APR-97	China
3	P00000003	Julia	F	16-DEC-65	Australia
4	P00000004	Benjamin	M	22-MAY-10	USA
5	P00000005	Ella	F	30-AUG-02	Germany
6	P00000006	Emma	F	19-OCT-10	Germany
7	P00000007	Lihua	M	19-OCT-93	China
8	P00000008	Edward	X	07-JUN-88	UK
9	P00000009	Iris	F	25-DEC-01	China

9 rows selected.

After: (Applicant #10 inserted)

```
SQL> SELECT * FROM Applicants;
```

APLCTID	PASSPORT	NAME	G	BIRTHDATE	COUNTRY
10	P12345678	David	M	12-JUL-99	China
1	P00000001	John	M	01-FEB-00	Spain
2	P00000002	Xiaohong	X	02-APR-97	China
3	P00000003	Julia	F	16-DEC-65	Australia
4	P00000004	Benjamin	M	22-MAY-10	USA
5	P00000005	Ella	F	30-AUG-02	Germany
6	P00000006	Emma	F	19-OCT-10	Germany
7	P00000007	Lihua	M	19-OCT-93	China
8	P00000008	Edward	X	07-JUN-88	UK
9	P00000009	Iris	F	25-DEC-01	China

10 rows selected.

2. DELETE(ON DELETE CASCADE) Operation: /aplct/withdraw.php line 34

DELETE FROM Applications WHERE appID = :appID

GUI:

* Application ID (required):

Your application has been withdrawn. Any visa issued will also be canceled.

Before:

```
SQL> SELECT * FROM Applications;
```

APPID	APLCTID	VOID	EID	RECEIVEDA	STATUS
1	1	1	1	19-JUN-17	Approved
2	2	2	2	20-SEP-18	Approved
3	3	3	3	21-DEC-15	Approved
4	4	4	4	22-JUL-18	Approved
5	5	5	5	23-SEP-20	Approved
6	1		1	07-OCT-19	Received
7	2		2	20-JAN-13	Received
8	3		3	11-NOV-19	Received
9	4		4	09-SEP-18	Received
10	5		5	19-JUL-17	Received
11	1		1	30-OCT-20	Received
12	2		2	01-SEP-21	Received
13	3		3	27-DEC-16	Received
14	4		4	18-SEP-19	Received
15	5	5	5	23-SEP-19	Received
16	2	2	2	23-SEP-19	Approved
17	2	2	2	24-SEP-19	Approved
18	2	2	2	25-SEP-19	Approved
19	2	2	2	26-SEP-19	Approved

19 rows selected.

```
SQL> SELECT * FROM Visas;
```

VID	APPID	APLCTID	VOID	ISSUEDATE	TY
1	1	1	1	15-NOV-20	T1
2	2	2	2	03-DEC-20	T1
3	3	2	3	22-OCT-22	T1
4	4	3	4	31-OCT-21	T1
5	5	5	5	18-NOV-20	T2
6	16	2	2	06-NOV-20	S1
7	17	2	2	05-NOV-20	S2
8	18	2	2	14-JAN-21	T2
9	19	2	2	18-MAR-22	W1

9 rows selected.

After: (Application #19 deleted so is Visa #9)

```
SQL> SELECT * FROM Applications;
```

APPID	APLCTID	VOID	EID	RECEIVEDA	STATUS
1	1	1	1	19-JUN-17	Approved
2	2	2	2	20-SEP-18	Approved
3	3	3	3	21-DEC-15	Approved
4	4	4	4	22-JUL-18	Approved
5	5	5	5	23-SEP-20	Approved
6	1		1	07-OCT-19	Received
7	2		2	20-JAN-13	Received
8	3		3	11-NOV-19	Received
9	4		4	09-SEP-18	Received
10	5		5	19-JUL-17	Received
11	1		1	30-OCT-20	Received
12	2		2	01-SEP-21	Received
13	3		3	27-DEC-16	Received
14	4		4	18-SEP-19	Received
15	5	5	5	23-SEP-19	Received
16	2	2	2	23-SEP-19	Approved
17	2	2	2	24-SEP-19	Approved
18	2	2	2	25-SEP-19	Approved

18 rows selected.

```
SQL> SELECT * FROM Visas;
```

VID	APPID	APLCTID	VOID	ISSUEDATE	TY
1	1	1	1	15-NOV-20	T1
2	2	2	2	03-DEC-20	T1
3	3	2	3	22-OCT-22	T1
4	4	3	4	31-OCT-21	T1
5	5	5	5	18-NOV-20	T2
6	16	2	2	06-NOV-20	S1
7	17	2	2	05-NOV-20	S2
8	18	2	2	14-JAN-21	T2

8 rows selected.

3. UPDATE Operation: /e/index.php line 76

UPDATE Applications SET void = :void WHERE applID = :applID

GUI:

Application ID: Visa Officer ID: [Assign](#)

Before:

```
SQL> SELECT * FROM Applications;
```

APPID	APLCTID	VOID	EID	RECEIVEDA	STATUS
1	1	1	1	19-JUN-17	Approved
2	2	2	2	20-SEP-18	Approved
3	3	3	3	21-DEC-15	Approved
4	4	4	4	22-JUL-18	Approved
5	5	5	5	23-SEP-20	Approved
6	1		1	07-OCT-19	Received
7	2		2	20-JAN-13	Received
8	3		3	11-NOV-19	Received
9	4		4	09-SEP-18	Received
10	5		5	19-JUL-17	Received
11	1		1	30-OCT-20	Received
APPID	APLCTID	VOID	EID	RECEIVEDA	STATUS
12	2		2	01-SEP-21	Received
13	3		3	27-DEC-16	Received
14	4		4	18-SEP-19	Received
15	5	5	5	23-SEP-19	Received
16	2	2	2	23-SEP-19	Approved
17	2	2	2	24-SEP-19	Approved
18	2	2	2	25-SEP-19	Approved
19	2	2	2	26-SEP-19	Approved

19 rows selected.

After (Application #7's void has been updated to 2):

```
SQL> SELECT * FROM Applications;
```

APPID	APLCTID	VOID	EID	RECEIVEDA	STATUS
1	1	1	1	19-JUN-17	Approved
2	2	2	2	20-SEP-18	Approved
3	3	3	3	21-DEC-15	Approved
4	4	4	4	22-JUL-18	Approved
5	5	5	5	23-SEP-20	Approved
6	1		1	07-OCT-19	Received
7	2	2	2	20-JAN-13	Received
8	3		3	11-NOV-19	Received
9	4		4	09-SEP-18	Received
10	5		5	19-JUL-17	Received
11	1		1	30-OCT-20	Received
APPID	APLCTID	VOID	EID	RECEIVEDA	STATUS
12	2		2	01-SEP-21	Received
13	3		3	27-DEC-16	Received
14	4		4	18-SEP-19	Received
15	5	5	5	23-SEP-19	Received
16	2	2	2	23-SEP-19	Approved
17	2	2	2	24-SEP-19	Approved
18	2	2	2	25-SEP-19	Approved
19	2	2	2	26-SEP-19	Approved

19 rows selected.

4. Selection: /e/index.php line 56

SELECT {\$attribute} FROM {\$table} WHERE eID = :eID

Sample output 1:

STATUS
Approved
Received
Received

Sample output 2:

NAME
Noah
Rachel

Applications/VO info at a glance:

Category: Info:

Embassy ID:

Applications/VO info at a glance:

Category: Info:

Embassy ID:

No change in the database.

5. Projection: /bo/index.php line 43

SELECT {\$attributesSelected} FROM Applicants

Sample output 1:

NAME	PASSPORT
John	P00000001
Xiaohong	P00000002
Julia	P00000003
Benjamin	P00000004
Ella	P00000005
Emma	P00000006
Lihua	P00000007
Edward	P00000008
Iris	P00000009

Sample output 2:

NAME	PASSPORT	GENDER	BIRTHDATE	COUNTRY
John	P00000001	M	01-FEB-00	Spain
Xiaohong	P00000002	X	02-APR-97	China
Julia	P00000003	F	16-DEC-65	Australia
Benjamin	P00000004	M	22-MAY-10	USA
Ella	P00000005	F	30-AUG-02	Germany
Emma	P00000006	F	19-OCT-10	Germany
Lihua	P00000007	M	19-OCT-93	China
Edward	P00000008	X	07-JUN-88	UK
Iris	P00000009	F	25-DEC-01	China

Info of all applicants

Please select attributes to include:

☒ name ☒ passport ☐ gender ☐ birthdate ☐ country

Show

Info of all applicants

Please select attributes to include:

☒ name ☒ passport ☒ gender ☒ birthdate ☒ country

Show

No change in the database.

6. Join: /vo/index.php line 42

```
SELECT A.appID, P.aplctID, P.name, P.passport, P.gender, P.birthDate, P.country, A.receiveDate  
FROM Applicants P, Applications A  
WHERE P.aplctID = A.aplctID AND A.status = 'Received' AND A.void = :void
```

Sample output:

New Applications

APPID	APLCTID	NAME	PASSPORT	GENDER	BIRTHDATE	COUNTRY	RECEIVEDATE
15	5	Ella	P00000005	F	30-AUG-02	Germany	23-SEP-19

Visa Officer ID: [Sign In](#)

No change in the database.

7. Aggregation with Group By: /index.php line 87

`SELECT country, COUNT(*) FROM Applicants GROUP BY country`

Sample output:

Aggregation with Group By: find the number of applicants for each country

COUNTRY	COUNT(*)
China	3
Australia	1
USA	1
Germany	2
UK	1
Spain	1

No change in the database.

8. Aggregation with Having: /index.php line 94

```
SELECT entryPoint, COUNT(*)  
FROM Admissions  
GROUP BY entryPoint  
HAVING COUNT(*) >= 2
```

Sample output:

Aggregation with Having:

Find the number of admissions for each entry point with at least 2 admissions

ENTRYPOINT	COUNT(*)
YVR	3
AKG	2
BLA	2

No change in the database.

9. Nested Aggregation with Group By: /index.php line 101

```
SELECT V.eID, SUM(V.experience)
FROM VisaOfficers V
GROUP BY V.eID
HAVING
    SUM(V.experience) >= ALL
    (SELECT SUM(V2.experience) FROM VisaOfficers V2 GROUP BY V2.eID)
```

Sample output:

Nested Aggregation with Group By:

find embassies whose sum of visa officer experiences is the maximum over all embassies

EID	SUM(V.EXPERIENCE)
2	19
5	19

No change in the database.

10. Division: /index.php line 107

```
SELECT void, name
FROM VisaOfficers O
WHERE NOT EXISTS
  ((SELECT T.type
    FROM VisaTypes T) MINUS
    (SELECT V.type
     FROM Visas V
     WHERE O.void = V.void))
```

Sample output:

Division: find all visa officers who have issued all types of visas

VOID	NAME
2	Noah

No change in the database.