University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: ____2____

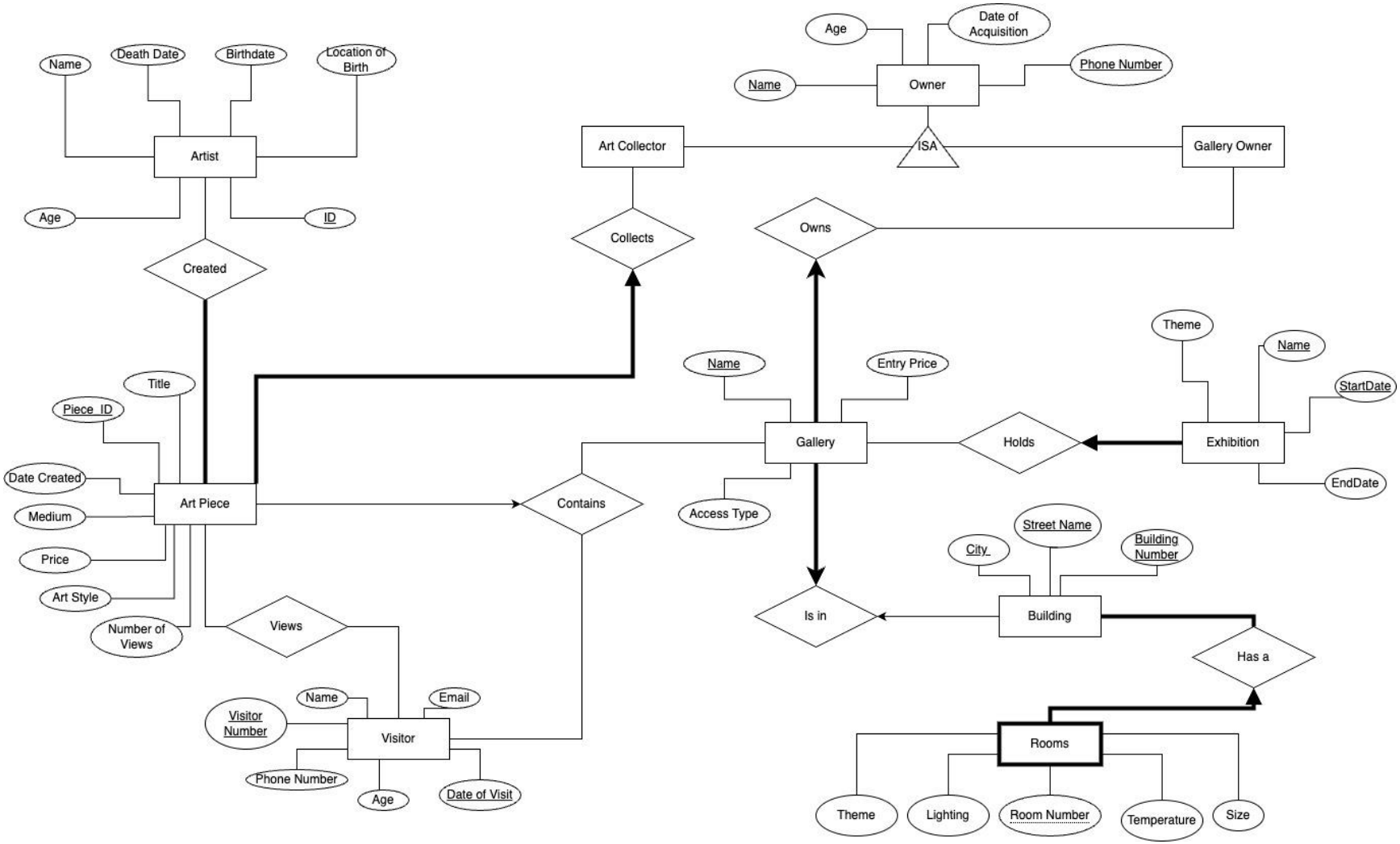
Date: _____October 17, 2022_____

Group Number: _____12_____

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Arianna Joe	85283554	a4o9r	aria231@student.ubc.ca
Tracy Chow	47273370	j2y3y	Tc2780@student.ubc.ca
Jason Huang	16005415	i4x2b	Jason.huang927@gmail.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



2. ER Diagram Changes:

- We added a one to many relationship between Building and Rooms as suggested by the TA on our feedback for Milestone 1

3. Relationship Schema derived from ER diagram

Table (ex: Table1(attr1: domain1, attr2: domain2, ...))	**Keys and Constraints (**primary key (PK), candidate key, (CK) foreign keys (FK), and other constraints)
Artist(Name: string, Age: number, DeathDate: Date, BirthDate: Date, LocationOfBirth: string, ID: number)	PK: ID CK: ID FK:
Created(**ArtistID**: number, **Piece_ID**: number)	PK: (**ArtistID, Piece_ID)** CK: (**ArtistID, Piece_ID)** FK: **ArtistID, Piece_ID** Constraints: art piece must have >=1 owner (artist created art piece)
Owner(Name: string, Age: number, DateOfAcquisition: Date, PhoneNumber: number)	PK: (Name, PhoneNumber) CK: (Name, PhoneNumber) FK: (art collector isa owner, gallery owner isa owner --- have 1 table with all attribute method)
Art Piece(Title: string, Piece_ID: number, DateCreated: Date, Medium: string, Price: number, ArtStyle: string, NumberOfViews: number, **CollectorName**: string, **CollectorPhoneNumber**: number, **GalleryName**: string)	PK: Piece_ID CK: Piece_ID FK: **CollectorName, CollectorPhoneNumber, GalleryName** (with gallery contains art piece, art collector collects piece)
Views(**VisitorNumber**: number, **DateOfVisit**: Date, **Piece_ID**: number)	PK: (**VisitorNumber, DateOfVisit, Piece_ID)** CK: (**VisitorNumber, DateOfVisit, Piece_ID)** FK: (**VisitorNumber, DateOfVisit, Piece_ID)** (visitor views art piece)
Visitor(Name: string, Email: string, Age: number, PhoneNumber: number, DateOfVisit: Date, VisitorNumber: number)	PK: (DateOfVisit, VisitorNumber) CK: (DateOfVisit, VisitorNumber) FK:
Contains(**VisitorNumber**: number, **DateOfVisit**: Date, **GalleryName**: string)	PK: (**VisitorNumber, DateOfVisit, GalleryName)** CK: (**VisitorNumber, DateOfVisit, GalleryName)** FK: **VisitorNumber, DateOfVisit** (gallery contains visitor)
Gallery(Name: string, EntryPrice: number, AccessType: string, **OwnerName**: string, **OwnerPhoneNumber**: number)	PK: (Name, **OwnerName, OwnerPhoneNumber)** CK: (Name, **OwnerName, OwnerPhoneNumber)** FK: **OwnerName, OwnerPhoneNumber**
IsIn(**GalleryName**: string, **City**: string,	PK: (**GalleryName, City, StreetName,**

StreetName: string, **BuildingNumber**: number)	**BuildingNumber)** CK: (**GalleryName, City, StreetName, BuildingNumber)** FK: **GalleryName, City, StreetName, BuildingNumber** (gallery is in building)
Exhibition(Theme: string, Name: string, StartDate: Date, EndDate: Date, **GalleryName**: string, **RoomTheme**: string)	PK: (Name, StartDate, **GalleryName)** CK: (Name, StartDate, **GalleryName)** FK: **GalleryName, RoomTheme**
Building(City: string, StreetName: string, BuildingNumber: number)	PK: City, StreetName, BuildingNumber CK: City, StreetName, BuildingNumber FK:
Rooms(Theme: string, Lighting: string, RoomNumber: number, Temperature: number, Size: number, **StreetName**: string, **BuildingNumber**: number, **City**: string)	PK: (RoomNumber, **StreetName, BuildingNumber, City)** CK: (RoomNumber, **StreetName, BuildingNumber, City)** FK: **StreetName, BuildingNumber, City**

4. Functional Dependencies (FDs)

Initial FDs from ER diagram:

- **Relations**
  - ArtPiece_ID -> Owner_Name, Owner_PhoneNumber
  - Gallery_Name -> Owner_Name, Owner_PhoneNumber
  - Gallery_Name -> Building_StreetName, Building_BuildingNumber, Building_City
  - Exhibition_StartDate, Exhibition_Name -> Gallery_Name
- **Tables**
  - Artist_ID -> Artist_Age, Artist_Name, Artist_DeathDate, Artist_BirthDate, Artist_LocationOfBirth
  - ArtPiece_ID -> ArtPiece_Title, ArtPiece_DateCreated, ArtPiece_Medium, ArtPiece_Price, ArtPiece_ArtStyle, ArtPiece_NumberOfViews
  - Visitor_VisitorNumber, Visitor_DateOfVisit -> Visitor_Name, Visitor_Email, Visitor_Phone Visitor_Number, Visitor_Age
  - Owner_Name, Owner_PhoneNumber -> Owner_Age, Owner_DateOfAcquisition
  - Gallery_Name -> Gallery_EntryPrice, Gallery_Accesstype
  - Exhibition_Name, Exhibition_startDate -> Exhibition_Theme, Exhibition_EndDate
  - BuildingNumber, Building_StreetName, Building_City -> Building_BuildingNumber, Building_StreetName, Building_City
  - Rooms_RoomNumber -> Rooms_Theme, Rooms_Lighting, Rooms_Temperature, Rooms_Size
- **Defined by Domain**
  - Exhibition_Theme -> Rooms_Theme, Rooms_Lighting
  - ArtPiece_Medium -> Rooms_Temperature
  - Artist_Birthdate, Artist_DeathDate -> Artist_Age

Attribute Legend:

Artist_ID	A.ID	Owner_Name	O.N
Artist_age	A.A	Owner_PhoneNumber	O.P
Artist_Name	A.N	Owner_Age	O.A
Artist_DeathDate	A.D	Owner_DateOfAcquisition	O.D
Artist_BirthDate	A.B	Gallery_Name	G.N
Aritst_LocationOfBirth	A.L	Gallery_AccessType	G.A
ArtPiece_ID	AP.ID	Gallery_EntryPrice	G.E
ArtPiece_Title	AP.T	Exhibition_Name	E.N
ArtPiece_DateCreated	AP.D	Exhibition_StartDate	E.S

ArtPiece_Medium	AP.M	Exhibition_EndDate	E.E
ArtPiece_Price	AP.P	Exhibition_Theme	E.T
ArtPiece_ArtStyle	AP.S	Building_City	B.C
ArtPiece_NumebrOfViews	AP.N	Building_StreetName	B.S
Visitor_VisitorNumber	V.ID	Building_BuildingNumber	B.B
Visitor_DateOfVisit	V.D	Rooms_RoomNumber	R.R
Visitor_Name	V.N	Rooms_Temperature	R.TE
Visitor_PhoneNumber	V.P	Rooms_Size	R.S
Visitor_Email	V.E	Rooms_Lighting	R.L
Visitor_Age	V.A	Rooms_Theme	R.TH

Translated FDs (and their closures)

FDs	Closure
AP.ID -> O.N, O.P	$AP.ID^+$ = {AP.ID, O.N, O.P, AP.T, AP.D, AP.M, AP.P, AP.S, AP.N, R.TE, O.A, O.D}
G.N -> O.N, O.P	$G.N^+$ = {G.N, O.N, O.P, B.C, B.B, B.S, G.A, G.E, O.A, O.D}
G.N -> B.C, B.B, B.S	$G.N^+$ = {G.N, B.C, B.B, B.S, O.N, O.P, G.A, G.E, O.A, O.D}
E.S, E.N -> G.N	E.S, $E.N^+$ = {E.S, E.N, G.N, O.N, O.P. B.C, B.B, B.S, G.A, G.E, O.A, O.D}
A.ID -> A.A, A.N, A.D, A.B, A.L	$A.ID^+$ = {A.ID, A.A, A.N, A.D, A.B, A.L}
AP.ID -> AP.T, AP.D, AP.M, AP.P, AP.S, AP.N	$AP.ID^+$ = {AP.ID, AP.T, AP.D, AP.M, AP.P, AP.S, AP.N, O.N, O.P, O.A, O.D, R.TE}
V.ID, V.D -> V.N, V.P, V.E, V.A	V.ID, $V.D^+$ = {V.ID, V.D, V.N, V.P, V.E, V.A}
O.N, O.P -> O.A, O.D	O.N, $O.P^+$ = {O.N, O.P, O.A, O.D}
G.N -> G.A, G.E	$G.N^+$ = {G.N, G.A, G.E, O.N, O.P, B.B, B.B, B.S, O.A, O.D}
E.N, E.S -> E.E, E.T	E.N, $E.S^+$ = {E.N, E.S, E.E, E.T, G.N, O.N, O.P. B.C, B.B, B.S, O.A, O.D, G.A, G.E, R.TH, R.L}

B.C, B.B, B.S-> B.C, B.B, B.S	B.C, B.B, B.S$^+$ = {B.C, B.B, B.S}
R.R -> R.TE, R.S, R.L, R.TH	R.R$^+$ = {R.R, R.TE, R.S, R.L, R.TH}
E.T -> R.TH, R.L	E.T$^+$ = {E.T, R.TH, R.L}
AP.M -> R.TE	AP.M$^+$ = {AP.M, R.TE}
A.B, A.D -> A.A	A.B, A.D$^+$ = {A.B, A.D, A.A}

Legend:

From Relations	From PKs	Defined by Domain

University of British Columbia, Vancouver
Department of Computer Science

5. Normalization

Finding Minimal Keys:

1.	~~AP.ID~~ ->	~~O.N, O.P~~
2.	~~G.N~~ ->	~~O.N, O.P~~
3.	~~G.N~~ ->	~~B.C, B.B, B.S~~
4.	~~E.S, E.N~~ ->	~~G.N~~
5.	~~A.ID~~ ->	A.A, A.N, ~~A.D, A.B~~, A.L
6.	~~AP.ID~~ ->	AP.T, AP.D, ~~AP.M~~, AP.P, AP.S, AP.N
7.	~~V.ID, V.D~~ ->	V.N, V.P, V.E, V.A
8.	~~O.N, O.P~~ ->	O.A, O.D
9.	~~G.N~~ ->	G.A, G.E
10.	~~E.N, E.S~~ ->	E.E, ~~E.T~~
11.	~~*B.C, B.B, B.S*~~ -> *trivial*	~~*B.C, B.B, B.S*~~ *trivial*
12.	~~R.R~~ , ~~B.C, B.S, B.B~~ ->	R.TE, R.S, R.L, R.TH
13.	~~E.T~~ ->	R.TH, R.L
14.	~~AP.M~~ ->	R.TE
15.	~~A.B, A.D~~ ->	A.A

Left	Middle	Right
AP.ID A.ID E.S E.N V.ID V.D R.R	G.N O.N O.P B.C B.B B.S E.T AP.M A.B A.D	A.A A.N A.L AP.T AP.D AP.P AP.S AP.N V.N V.P V.E V.A

		O.A O.D G.A G.E E.E R.TE R.S R.L R.TH

Minimal Keys:

AP.ID, A.ID, E.S, E.N, V.ID, V.D, $R.R^+$ = {

A.ID, A.A, A.N, A.D, A.B, A.L,
AP.ID, AP.T, AP.D, AP.M, AP.P, AP.S, AP.N,
V.ID, V.D, V.N, V.P, V.E, V.A,
O.N, O.P, O.A, O.D,
G.N, G.A, G.E,
E.N, E.S, E.E, E.T,
B.C, B.S, B.B,
R.R, R.TE, R.S, R.L, R.TH

}

^ covers all attributes from legend → min keys = {AP.ID, A.ID, E.S, E.N, V.ID, V.D, R.R}

Primary keys not in min keys = {O.N, O.P, G.N, B.C, B.S, B.B}

→ Not in 3NF: many violations where given X -> b, b is not in min keys for multiple defined FDs

Step 1- Standard Form:

AP.ID ->	O.N	V.ID, V.D ->	V.P
AP.ID ->	O.P	V.ID, V.D ->	V.E
G.N ->	O.N	V.ID, V.D ->	V.A
G.N ->	O.P	O.N, O.P ->	O.A
G.N ->	B.C	O.N, O.P ->	O.D

G.N ->	B.B	G.N ->	G.A
G.N ->	B.S	G.N ->	G.E
E.S, E.N ->	G.N	E.N, E.S ->	E.E
A.ID ->	A.A	E.N, E.S ->	E.T
A.ID ->	A.N	*B.C, B.B, B.S* -> *trivial*	*B.C* *trivial*
A.ID ->	A.D	*B.C, B.B, B.S* -> *trivial*	*B.B* *trivial*
A.ID ->	A.B	*B.C, B.B, B.S* -> *trivial*	*B.S* *trivial*
A.ID ->	A.L	R.R, B.C, B.S, B.B ->	R.TE
AP.ID ->	AP.T	R.R, B.C, B.S, B.B ->	R.S
AP.ID ->	AP.D	R.R, B.C, B.S, B.B ->	R.L
AP.ID ->	AP.M	R.R, B.C, B.S, B.B ->	R.TH
AP.ID ->	AP.P	E.T ->	R.TH
AP.ID ->	AP.S	E.T ->	R.L
AP.ID ->	AP.N	AP.M ->	R.TE
V.ID, V.D ->	V.N	A.B, A.D ->	A.A

Step 2 - Reduce LHS:

$E.S^+$ = {E.S, G.N, O.N, O.P, B.C, B.S, B.B} ✘

$E.N^+$ = {E.N, G.N, O.N, O.P, B.C, B.S, B.B} ✘

$V.ID^+$ = {V.ID, V.N, V.P, V.E} ✘

$V.D^+$ = {V.D, V.N, V.P, V.E} ✘

…same for all FDs with more than 1 attribute on LHS

→ All already in min

Step 3 - Remove Redundant FDs:

~Removed trivial FDs~

G.N ->	O.N	V.ID, V.D ->	V.N
G.N ->	O.P	V.ID, V.D ->	V.P

G.N ->	B.C	V.ID, V.D ->	V.E
G.N ->	B.B	V.ID, V.D ->	V.A
G.N ->	B.S	O.N, O.P ->	O.A
G.N ->	G.A	O.N, O.P ->	O.D
G.N ->	G.E	E.N, E.S ->	G.N
A.ID ->	A.A	E.N, E.S ->	E.E
A.ID ->	A.N	E.N, E.S ->	E.T
A.ID ->	A.D	E.T ->	R.TH
A.ID ->	A.B	E.T ->	R.L
A.ID ->	A.L	~~*B.C, B.B, B.S ->* *trivial*~~	~~*B.C* *trivial*~~
AP.ID ->	O.N	~~*B.C, B.B, B.S ->* *trivial*~~	~~*B.B* *trivial*~~
AP.ID ->	O.P	~~*B.C, B.B, B.S ->* *trivial*~~	~~*B.S* *trivial*~~
AP.ID ->	AP.T	R.R, B.C, B.S, B.B ->	R.TE
AP.ID ->	AP.D	R.R, B.C, B.S, B.B ->	R.S
AP.ID ->	AP.M	R.R, B.C, B.S, B.B ->	R.L
AP.ID ->	AP.P	R.R, B.C, B.S, B.B ->	R.TH
AP.ID ->	AP.S	AP.M ->	R.TE
AP.ID ->	AP.N	A.B, A.D ->	A.A

Tables After Step 3:

G.N ->	O.N, O.P, B.C, B.B, B.S, G.A, G.E	V.ID, V.D ->	V.N, V.P, V.E, V.A
A.ID ->	A.A, A.N, A.D, A.B, A.L	O.N, O.P ->	O.A, O.D
AP.ID ->	O.N, O.P, .AP.T, AP.D, AP.M, AP.P, AP.S,	E.N, E.S ->	G.N, E.E, E.T

	AP.N		
AP.M ->	R.TE	E.T ->	R.TH, R.L
A.B, A.D ->	A.A	R.R, B.C, B.S, B.B ->	R.TE, R.S, R.L, R.TH

Decomposition:

min keys = {AP.ID, A.ID, E.S, E.N, V.ID, V.D, R.R}

INITIAL DECOMPOSITION		**ADDITIONAL RELATION(S)**
R1(G.N, O.N, O.P, B.C, B.B, B.S, G.A, G.E)	R6(**V.ID**, **V.D**, V.N, V.P, V.E, V.A)	R11(**AP.ID, A.ID, E.S, E.N, V.ID, V.D, R.R**) *~no relation (R1-R10) has all keys; so add in relation w/ all minimal keys~*
R2(**A.ID**, A.A, A.N, A.D, A.B, A.L)	R7(O.N, O.P, O.A, O.D)	
R3(**AP.ID**, O.N, O.P, .AP.T, AP.D, AP.M, AP.P, AP.S, AP.N)	R8(**E.N**, **E.S**, G.N, E.E, E.T)	
R4(AP.M, R.TE)	R9(E.T, R.TH, R.L)	
~~R5(A.B, A.D, A.A)~~ *~duplicate to R2; so remove redundant relation~*	R10(**R.R**, B.C, B.S, B.B, R.TE, R.S, R.L, R.TH)	

FINAL RELATIONS	
R1(G.N, O.N, O.P, B.C, B.B, B.S, G.A, G.E)	R7(O.N, O.P, O.A, O.D)
R2(**A.ID**, A.A, A.N, A.D, A.B, A.L)	R8(**E.N**, **E.S**, G.N, E.E, E.T)
R3(**AP.ID**, O.N, O.P, .AP.T, AP.D, AP.M, AP.P, AP.S, AP.N)	R9(E.T, R.TH, R.L)
R4(AP.M, R.TE)	R10(**R.R**, B.C, B.S, B.B, R.TE, R.S, R.L, R.TH)

R6(**V.ID**, **V.D**, V.N, V.P, V.E, V.A)	R11(**AP.ID, A.ID, E.S, E.N, V.ID, V.D, R.R**) (add in table w/ all minimal keys)

Tables after normalization:

Corresponding derived table from decomposition	**Table** (ex: Table1(attr1: domain1, attr2: domain2, ...))	**Keys and Constraints (**primary key (PK), candidate key, (CK) foreign keys (FK), and other constraints)
R2	Artist(Name: string, Age: number, DeathDate: Date, BirthDate: Date, LocationOfBirth: string, ID: number)	PK: ID CK: ID FK:
R11	Created(**Artist_ID**: number, **ArtPiece_ID**: number, **Exhibition_StartDate**: Date, **Exhibition_Name**: string, **Visitor_ID**: number, **Visitor_DateOfVisit**: Date, **Rooms_RoomNumber**: number)	PK: (**Artist_ID, ArtPiece_ID, Exhibition_StartDate, Exhibition_Name, Visitor_ID, Visitor_DateOfVisit, Rooms_RoomNumber)** CK: (**Artist_ID, ArtPiece_ID, Exhibition_StartDate, Exhibition_Name, Visitor_ID, Visitor_DateOfVisit, Rooms_RoomNumber)** FK: **Artist_ID, ArtPiece_ID, Exhibition_StartDate, Exhibition_Name, Visitor_ID, Visitor_DateOfVisit, Rooms_RoomNumber** Constraints: art piece must have >=1 owner (artist created art piece)
R7	Owner(Owner_Name: string, Owner_Age: number, Owner_DateOfAcquisition: Date, Owner_PhoneNumber: number)	PK: (Owner_Name, Owner_PhoneNumber) CK: (Owner_Name, Owner_PhoneNumber) FK: (art collector isa owner, gallery owner isa owner)
R3	Art Piece(ArtPiece_Title: string, ArtPiece_ID: number, ArtPiece_DateCreated: Date, ArtPiece_Medium: string, ArtPiece_Price: number, ArtPiece_ArtStyle: string, **Owner_Name**: string, **Owner_PhoneNumber**: number, **Gallery_Name**: string)	PK: ArtPiece_ID CK: ArtPiece_ID FK: **Owner_Name, Owner_PhoneNumber, Gallery_Name** (with gallery contains art piece,

		art collector collects piece)
R1	GalleryInformation(Gallery_Name: string, **Owner_Name**: string, **Owner_PhoneNumber**: number, Building_City: string, Building_StreetName: string, Building_BuildingNumber: number, Gallery_AccessType: string, Gallery_EntryPrice: number)	PK: Gallery_Name CK: Gallery_Name FK: **Owner_Name, Owner_PhoneNumber**
R6	Visitor(Visitor_Name: string, Visitor_Email: string, Visitor_Age: number, Visitor_PhoneNumber: number, Visitor_DateOfVisit: Date, Visitor_Number: number)	PK: (Visitor_DateOfVisit, Visitor_Number) CK: (Visitor_DateOfVisit, Visitor_Number) FK:
R4	ArtPieceRoom(**ArtPiece_Medium**: string, **Rooms_Temperature**: number)	PK: **ArtPiece_Medium** CK: **ArtPiece_Medium** FK: **ArtPiece_Medium, Rooms_Temperature**
R8	Exhibition(Exhibition_Theme: string, Exhibition_Name: string, Exhibition_StartDate: Date, Exhibition_EndDate: Date, **Gallery_Name**: string)	PK: (Exhibition_Name, Exhibition_StartDate, **Gallery_Name)** CK: (Exhibition_Name, Exhibition_StartDate, **Gallery_Name)** FK: **Gallery_Name**
R9	ExhibitionRoomSettings(**Exhibition_Theme**: string, **Rooms_Theme**: string, **Rooms_Lighting**: string)	PK: **Exhibition_Theme** CK: **Exhibition_Theme** FK:-**Exhibition_Theme, Rooms_Theme, Rooms_Lighting**
R10	Rooms(Rooms_Theme: string, Rooms_Lighting: string, Rooms_RoomNumber: number, Rooms_Temperature: number, Rooms_Size: number, **Building_StreetName**: string, **Building_BuildingNumber**: number, **Building_City**: string)	PK: (Rooms_RoomNumber, **Building_StreetName, Building_BuildingNumber, Buildling_City)** CK: (Rooms_RoomNumber, **Building_StreetName, Building_BuildingNumber, Building_City)** FK: **Building_StreetName, Building_BuildingNumber, Building_City**

6. SQL DDL TABLES

FINAL RELATIONS	
R1(G.N, O.N, O.P, B.C, B.B, B.S, G.A, G.E)	R7(O.N, O.P, O.A, O.D)
R2(**A.ID**, A.A, A.N, A.D, A.B, A.L)	R8(**E.N**, **E.S**, G.N, E.E, E.T)
R3(**AP.ID**, O.N, O.P, .AP.T, AP.D, AP.M, AP.P, AP.S, AP.N)	R9(E.T, R.TH, R.L)
R4(AP.M, R.TE)	R10(**R.R**, B.C, B.S, B.B, R.TE, R.S, R.L, R.TH)
R6(**V.ID**, **V.D**, V.N, V.P, V.E, V.A)	R11(**AP.ID, A.ID, E.S, E.N, V.ID, V.D, R.R**) (add in table w/ all minimal keys)

R1(G.N, O.N, O.P, B.C, B.B, B.S, G.A, G.E)

```
CREATE TABLE Gallery(
        Gallery_Name            CHAR(20) PRIMARY KEY,
        Gallery_AccessType      CHAR(20),
        Gallery_EntryPrice      NUMBER,
        Owner_Name              CHAR(20),
        Owner_PhoneNumber       INTEGER,
        Building_City           CHAR(20),
        Building_BuildingNumber INTEGER,
        Building_StreetName     CHAR(20)

        FOREIGN KEY (Owner_Name,Owner_PhoneNumber) REFERENCES
                Owner(Owner_Name,Owner_PhoneNumber)
                ON DELETE SET NULL
                ON UPDATE CASCADE
        UNIQUE (Building_City, Building_BuildingNumber, Building_StreetName)
)
```

R2(A.ID, A.A, A.N, A.D, A.B, A.L)

```
CREATE TABLE Artist(
        Artist_ID               INTEGER PRIMARY KEY,
        Artist_Age              INTEGER,
        Artist_Name             CHAR(20),
        Artist_DeathDate        DATE,
        Artist_BirthDate        DATE,
        Artist_LocationOfBirth  CHAR(30) !!! GEO???
)
```

R3(AP.ID, O.N, O.P, AP.T, AP.D, AP.M, AP.P, AP.S, AP.N)

```
CREATE TABLE ArtPiece(
        ArtPiece_ID             INTEGER PRIMARY KEY,
        ArtPiece_Title          CHAR(20),
        ArtPiece_DateCreated    DATE,
        ArtPiece_Medium         CHAR(20),
        ArtPiece_Price          NUMBER,
        ArtPiece_Style          CHAR(20),
        ArtPiece_NumberOfViews  INTEGER,
        Owner_Name              CHAR(20),
        Owner_PhoneNumber       INTEGER,
        Gallery_Name            CHAR(20),

        FOREIGN KEY (Owner_Name,Owner_PhoneNumber) REFERENCES
                Owner(Owner_Name,Owner_PhoneNumber)
                ON DELETE SET NULL
                ON UPDATE CASCADE

        FOREIGN KEY (Gallery_Name) REFERENCES Gallery(Gallery_Name)
                ON DELETE SET NULL
                ON UPDATE CASCADE
)
```

R4(AP.M, R.TE)

```
CREATE TABLE ArtPieceRoom(
        ArtPiece_Medium         CHAR(30) PRIMARY KEY,
        Rooms_Temperature       NUMBER,

        FOREIGN KEY (ArtPiece_Medium) REFERENCES ArtPiece(ArtPiece_Medium)
        FOREIGN KEY (Room_Temperature) REFERENCES Rooms(Room_Temperature)
)
```

R6(V.ID, V.D, V.N, V.P, V.E, V.A)

```
CREATE TABLE Visitor(
        Visitor_ID              INTEGER,
        Visitor_DateOfVisit     DATE,
        Visitor_Name            CHAR(20),
        Visitor_PhoneNumber     INTEGER,
        Visitor_Email           CHAR(30),
        Visitor_Age             INTEGER,

        PRIMARY KEY (Visitor_ID,Visitor_DateOfVisit)
)
```

R7(O.N, O.P, O.A, O.D)

```
CREATE TABLE Owner(
        Owner_Name              CHAR(20),
        Owner_PhoneNumber       INTEGER,
        Owner_Age               INTEGER,
        Owner_DateOfAcquisition DATE,

        PRIMARY KEY (Owner_Name, Owner_PhoneNumber)
)
```

R8(E.N, E.S, G.N, E.E, E.T)

```
CREATE TABLE Exhibition(
        Exhibition_Name         CHAR(30),
        Exhibition_StartDate    DATE,
        Exhibition_EndDate      DATE,
        Exhibition_Theme        CHAR(20),
        Gallery_Name            CHAR(20),

        PRIMARY KEY (Exhibition_Name,Exhibition_StartDate),
        FOREIGN KEY (Gallery_Name) REFERENCES Gallery(Gallery_Name)
                ON DELETE SET NULL
                ON UPDATE CASCADE
)
```

R9(E.T, R.TH, R.L)

```
CREATE TABLE ExhibitionRoomSettings(
        Exhibition_Theme        CHAR(20) PRIMARY KEY,
        Rooms_Lighting          CHAR(20),
        Rooms_Theme             CHAR(30)

        FOREIGN KEY (Exhibition_Theme) REFERENCES Exhibition(Exhibition_Theme)
        FOREIGN KEY (Rooms_Lighting, Rooms_Theme) REFERENCES
                Rooms(Rooms_Lighting, Rooms_Theme)
)
```

R10(R.R, B.C, B.S, B.B, R.TE, R.S, R.L, R.TH)

```
CREATE TABLE Rooms(
        Rooms_RoomNumber        CHAR(20),
        Rooms_Temperature       NUMBER,
        Rooms_Size              NUMBER,
        Rooms_Lighting          CHAR(20)
        Rooms_Theme             CHAR(30),
        Building_City           CHAR(20),
        Building_StreetName     CHAR(30),
```

```
        Building_BuildingNumber      INTEGER,

        PRIMARY KEY (Building_City, Building_BuildingNumber, Building_StreetName,
        Rooms_RoomNumber)
        FOREIGN KEY (Building_City, Building_BuildingNumber,Building_StreetName)
        REFERENCES Gallery(Building_City, Building_BuildingNumber, Building_StreetName)
                ON DELETE CASCADE
                ON UPDATE CASCADE
)
```

R11(AP.ID, A.ID, E.S, E.N, V.ID, V.D, R.R)

```
CREATE TABLE Created(
        ArtPiece_ID                 INTEGER
        Artist_ID                   INTEGER
        Exhibition_Name             CHAR(30)
        Exhibition_StartDate        DATE
        Visitor_VisitorNumber       INTEGER
        Visitor_DateOfVisit         DATE
        Rooms_RoomNumber            INTEGER

        PRIMARY KEY (ArtPiece_ID, Artist_ID, Exhibition_Name, Exhibition_StartDate,
                Visitor_VisitorNumber, Visitor_DateOfVisit, Rooms_RoomNumber)

        FOREIGN KEY (ArtPiece_ID) REFERENCES ArtPiece(ArtPiece_ID)
                ON DELETE CASCADE
                ON UPDATE CASCADE

        FOREIGN KEY (Artist_ID) REFERENCES Artist(Artist_ID)
                ON DELETE CASCADE
                ON UPDATE CASCADE

        FOREIGN KEY (Exhibition_Name, Exhibition_StartDate) REFERENCES
                Exhibition(Exhibition_Name, Exhibition_StartDate)
                ON DELETE CASCADE
                ON UPDATE CASCADE

        FOREIGN KEY (Visitor_VisitorNumber, Visitor_DateOfVisit) REFERENCES
                Visitor(Visitor_VisitorNumber, Visitor_DateOfVisit)
                ON DELETE CASCADE
                ON UPDATE CASCADE

        FOREIGN KEY (Rooms_RoomNumber) REFERENCES
                Rooms(Rooms_RoomNumber)
                ON DELETE CASCADE
                ON UPDATE CASCADE
)
```

University of British Columbia, Vancouver
Department of Computer Science

7. SQL INSERT TUPLES

GALLERY

```
INSERT INTO Gallery (Gallery_Name, Gallery_AccessType,Gallery_EntryPrice, Owner_Name,
    Owner_PhoneNumber, Building_City, Building_BuildingNumber, Building_StreetName)

    VALUES ("Decora", "Public", 55, "Bobby", 41633454354, "Toronto", 1234, "Dundas"),
        ("Antiquities", "Private", 650, "Norman", 41643454754, "Vancouver", 232,
            "Manor"),
        ("Artovox", "Public", 750, "Nicole", 60443454654, "Winnipeg", 23312,
            "Walter Main"),
        ("Mural", "Public", 70, "Summer", 64743454614, "Calgary", 1212, "Scurfield"),
        ("Patronizing", "Private", 1170, "Dion", 2896564783, "Toronto", 2360, "Younge")
```

ARTIST

```
INSERT INTO Artist(Artist_ID, Artist_Age, Artist_Name, Artist_DeathDate, Artist_BirthDate,
    Artist_LocationOfBirth)

    VALUES (1111, 64, "Bobby Brown", 2019-04-09, 1954-05-12, "Toronto"),
        (1112, 50, "Charlie Red", 2020-05-19, 1970-01-11, "Mexico"),
        (1113, 43, "Donald Dan", 1995-04-01, 1952-02-17, "Shanghai"),
        (1114, 106, "Daisy Dry", 2022-10-01, 1916-01-01, "Vancouver"),
        (1115, 30, "Mickey Mountain", 2008-10-01, 1978-09-12, "Calgary")
```

ART PIECE

```
INSERT INTO ArtPiece(ArtPiece_ID, ArtPiece_Title, ArtPiece_DateCreated, ArtPiece_Medium,
    ArtPiece_Price, ArtPiece_Style, ArtPiece_NumberOfViews, Owner_Name,
    Owner_PhoneNumber, Gallery_Name)

    VALUES (10, "Mona Lisa", 1503-09-27, "Oil", 100000, "Human Portrait", 123455432,
            "Bobby", 41633454354, "Decora"),
        (11, "Starry Nights", 1490-01-12, "Water colour", 1338484, "Art Deco",
            6367488, "Norman", 41643454754, "Antiquities"),
        (12, "Mouse in the Trap", 1990-12-01, "Acrylics", 348990, "Baroque", 839829,
            "Summer", 60443454654, "Artovox"),
        (13, "Duck Duck Goose", 1998-08-27, "Chalk", 3498394839, "Bauhaus",
            29302803, "Nicole", 64743454614, "Mural"),
        (14, "NeonRed In Your Area", 2014-04-06, "Charcoal", 34938943,
            "Art Nouveau", 93843, "Dion", 2896564783, "Patronizing")
```

ART PIECE ROOM

INSERT INTO ArtPieceRoom(ArtPiece_Medium, Room_Temperature)
VALUES ("Oil", 23,4), ("Water colour", 24.3), ("Acrylics", 24.3), ("Chalk", 23.4),
("Charcoal", 25.1)

VISITOR

INSERT INTO Visitor(Visitor_ID, Visitor_DateOfVisit, Visitor_Name,
Visitor_PhoneNumber, Visitor_Email, Visitor_Age)

VALUES (1, 2022-10-10, "Alan", 4161111111, "alan.apple@gmail.com", 45),
(2, 2022-05-05, "Bob", 4162222222, "bob.banana@gmail.com", 13)
(3, 2022-07-10, "Cameron", 4162322323,
"cameron.cherrie@gmail.com", 61)
(4, 2022-08-11, "Daniel", 4163233322, "daniel.date@gmail.com", 62)
(5, 2022-02-14, "Ethan", 4161144141, "ethan.eggplant@gmail.com", 65)
(6, 2022-10-13, "Frankie", 4161233213, "frankie.fruit@gmail.com", 12)

OWNER

INSERT INTO Owner(Owner_Name, Owner_PhoneNumber, Owner_Age,
Owner_DateOfAcquisition)

VALUES ("Bobby", 41633454354, 56, 2019-05-05),
("Norman", 41643454754, 54, 2018-04-15),
("Summer", 60443454654, 78, 2022-09-05),
("Nicole", 64743454614, 88, 2012-01-05),
("Dion", 2896564783, 10, 2004-06-05)

EXHIBITION

INSERT INTO Exhibition(Exhibition_Name, Exhibition_StartDate, Exhibition_EndDate,
Exhibition_Theme, Gallery_Name)

VALUES ("DoomsDay", 2022-10-01, 2022-12-01, "Wine & Dine", "Decora"),
("SoupDay", 2021-05-01, 2021-08-01, "Cheese with a side of Pizza",
"Antiquities"),
("YAHOO", 2014-06-01, 2014-12-01, "Casual Date", "Artovox"),
("BOOMBAYAH", 2017-08-01, 2017-09-01, "Black with a dash of Pink",
"Mural"),
("Love Dive", 2022-02-01, 2022-04-01, "Be Narcissistic", "Patronizing")

EXHIBITION ROOM SETTINGS

INSERT INTO ExhibitionRoomSettings(Exhibition_Theme, Rooms_Lighting, Rooms_Theme)

VALUES ("Wine & Dine", "Dim lights", "Gloomy"),
("SoupDay", "Bright lights", "Rainy with a chance of spaghetti"),
("YAHOO", "Ambient lights", "Midwest Vibes"),
("BOOMBAYAH", "Neon lights", "DDU DU DDU DU"),
("Love Dive", "Accent lights", "Flying Cars")

ROOMS

R10(R.R, B.C, B.S, B.B, R.TE, R.S, R.L, R.TH)

INSERT INTO Rooms(Rooms_RoomNumber, Rooms_Temperature, Rooms_Size, Rooms_Lighting, Rooms_Theme, Building_City, Building_StreetName, Building_BuildingNumber)

VALUES (1, 23.4, "Small", "Dim lights", "Gloomy", "Toronto", 1234, "Dundas"),
(2, 24.3, "Medium", "Bright lights", "Rainy with a chance of spaghetti", "Vancouver", 232, "Manor"),
(3, 24.3, "Large", "Ambient lights", "Midwest Vibes", "Winnipeg", 23312, "Walter Main"),
(4, 23.4, "Small", "Neon lights", "DDU DU DDU DU", "Calgary", 1212, "Scurfield"),
(5, 25.1, "Small", "Accent lights", "Flying Cars", "Toronto", 2360, "Younge")

CREATED

INSERT INTO CREATED (ArtPiece_ID, Artist_ID, Exhibition_Name, Exhibition_StartDate, Visitor_VisitorNumber, Visitor_DateOfVisit, Rooms_RoomNumber)

VALUES (10, 1111, "DoomsDay", 2022-10-01, 1, 2022-10-10, 1),
(11, 1112, "SoupDay", 2021-05-01, 2, 2021-05-05, 2),
(12, 1113, "YAHOO", 2014-06-01, 3, 2014-07-10, 3),
(13, 1114, "BOOMBAYAH", 2017-08-01, 4, 2017-08-11, 4),
(14, 1115, "Love Dive", 2022-02-01, 5, 2022-02-14, 5)

University of British Columbia, Vancouver
Department of Computer Science

Deliverables

Each group must provide the following as a **single** PDF file:

1. ~~A completed cover page (template on Canvas)~~
2. ~~The ER diagram you are basing your item #3 (below) on. This ER diagram may be the same as your milestone 1 submission or it might be different.~~ If you have made changes from the version submitted in milestone 1, attach a note indicating what changes have been made and why.
3. The schema derived from your ER diagram (above). For the translation of the ER diagram to the relational model, follow the same instructions as in your lectures. The process should be reasonably straightforward. For each table:
   a. List the table definition (e.g., Table1(attr1: domain1, attr2: domain2, ...))

   b. Specify the primary key (PK), candidate key, (CK) foreign keys (FK), and other constraints that the table must maintain.

4. Functional Dependencies (FDs)

   a. Identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key).

   Note: In your list of FDs, there must be some kind of valid FD other those identified by a PK or CK. If you observe that no relations have FDs other than the PK and CK(s), then you will have to intentionally add some (meaningful) attributes to show valid FDs. We want you to get a good normalization exercise. Your design must go through a normalization process.

5. Normalization

   a. Normalize each of your tables to be in 3NF or BCNF. Give the list of tables, their primary keys, their candidate keys, and their foreign keys after normalization.

   You should show the steps taken for the decomposition. Should there be errors, and no work is shown, no partial credit can be awarded without steps shown. The format should be the same as Step 3, with tables listed similar to Table1(attr1:domain1, attr2:domain2, ...). ALL Tables must be listed, not only the ones post normalization.

6. The SQL DDL statements required to create all the tables from item #5. The statements should use the appropriate foreign keys, primary keys, UNIQUE constraints, etc.

7. INSERT statements to populate each table with at least 5 tuples. You will likely want to have more than 5 tuples so that you can have meaningful queries later on.