University of British Columbia, Department of Computer Science

CPSC 304

Winter 2017 Term 2

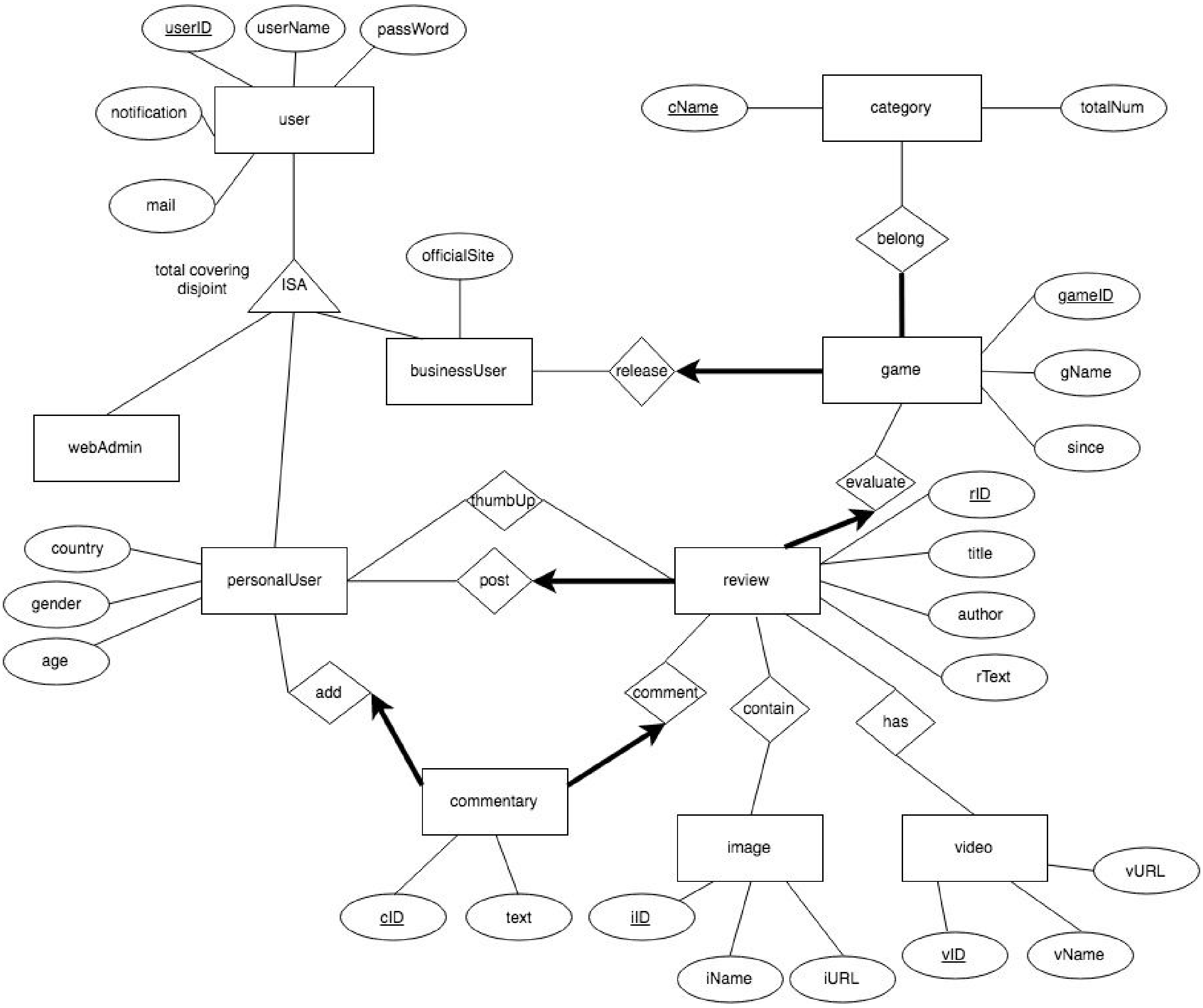
Project Part 2: E/R Diagrams and Schemas

Group Members:

Name	Student Number	Unix ID	Tutorial Section	Email Address
Hanbo Yang	55683163	u3x0b	T2E	kirigiricloud@gmail.com
Tongtong Zhai	29108164	s5b1b	T2G	darcy.tongtong.zhai@gmail.com
Yilan Yan	34914168	h2g1b	T2G	frankyan1129@gmail.com
Zhuo Yang	52411162	p2e1b	T2G	zhuoyang0912@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 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

ER Diagram:



Schemas:

Notification is an enum: { 0 = Not Notified, 1 = Notified }.

Commentary (cID: Int, text: String, userID: Int, rID: Int)

Primary Keys: cID
Candidate Keys: None
Foreign Keys: userID references PersonalUser, rID references Review

Functional Dependencies:

cID → {text, userID, rID}
cID is the Commentmentary ID, the primary key of the table. The primary key implies the rest of the attributes.

This table is in BCNF because cID is a super key of this table.

SQL DDL

```
CREATE TABLE Commentary
(
      cID INT,
      text CHAR(200),
      userID INT NOT NULL,
      rID INT NOT NULL,
      PRIMARY KEY (cID),
      FOREIGN KEY (userID) REFERENCES PersonalUser(userID)
            ON DELETE CASCADE
            ON UPDATE CASCADE,
      FOREIGN KEY (rID) REFERENCES Review(rID)
            ON DELETE CASCADE
            ON UPDATE CASCADE
);
```

Example 5 Tuples:

cID	text	userID	rID
1	Yeah, I totally agree with you!	1	1
2	Blah, blah, blah...	2	2
3	Blah, blah, blah...	3	2
4	Blah, blah, blah...	4	3
5	Blah, blah, blah...	5	4

Review (rID: Int, title: String, author: String, text: String, userID: Int, gameID: Int)

Primary Keys: rID
Candidate Keys: None
Foreign Keys: userID references PersonalUser, gameID references Game

Functional Dependencies:

rID → {title, author, text, userID, gameID}
rID is the Review ID, the primary key of the table. The primary key implies the rest of the attributes.

This table is in BCNF because rID is a super key of this table.

SQL DDL

```
CREATE TABLE Review
(
      rID INT,
      title CHAR(100),
      author CHAR(20),
      text CHAR(2000),
      userID INT NOT NULL,
      gameID INT NOT NULL,
      PRIMARY KEY (rID),
      FOREIGN KEY (userID) REFERENCES PersonalUser(userID)
            ON DELETE CASCADE
            ON UPDATE CASCADE
      FOREIGN KEY (gameID) REFERENCES Game(gameID)
            ON DELETE CASCADE
            ON UPDATE CASCADE
);
```

Example 5 Tuples:

rID	title	author	text	userID	gameID
1	Dynasty Warrior Review	Ashley	I LOVE this game!!!	11	1
2	CS GO Review	Blake	A very high quality game! Blah, blah...	12	2
3	GT Sport Review	Cris	Blah, blah, blah...	13	3

4	NBA 2k18 Review	David	Blah, blah, blah...	14	4
5	Super Mario Review	Eddie	Blah, blah, blah...	15	5

Image (iID: Int, iName: String, iURL: String)

Primary Keys: iID
Candidate Keys: iURL
Foreign Keys: None

Functional Dependencies:

iID → {iName, iURL}
iID is the Image ID, the primary key of the table. The primary key implies the rest of the attributes.

iURL → {iName, iID}
iURL is a candidate key of table. This also implies all attributes of image.

This table is in BCNF because in both FDs, iID and iURL are super keys of this table.

SQL DDL

```
CREATE TABLE Image
(
      iID INT,
      iName CHAR(100) NOT NULL,
      iURL CHAR(200) NOT NULL,
      PRIMARY KEY (iID),
      UNIQUE(iURL)
);
```

Example 5 Tuples:

iID	iName	iURL
100	2017-Top-Ten-Game.jpg	https://games4fun.com/2017-Top-Ten-Game.jpg
200	NBA-2k18-Cover-a.jpg	https://games4fun.com/NBA-2k18-Cover-a.jpg
300	GO-CS-Trial.gif	https://games4fun.com/GO-CS-Trial.gif
400	Super-Mario-View.jpg	https://games4fun.com/Super-Mario-View.jpg

500	web-public-title-red.jpg	https://games4fun.com/web-public-title-red.jpg

Video (vID: Int, vName: String, vURL: String)

Primary Keys: iID
Candidate Keys: vURL
Foreign Keys: None

Functional Dependencies:

vID → {vName, vURL}
vID is the Video ID, the primary key of the table. The primary key implies the rest of the attributes.

vURL → {vName, vID}
IURL is a candidate key of table. This also implies all attributes of image.

This table is in BCNF because both vID and vURL are super keys of this table.

SQL DDL

```
CREATE TABLE Video
(
     vID INT,
     vName CHAR(100)NOT NULL,
     vURL CHAR(200) NOT NULL,
     PRIMARY KEY (vID),
     UNIQUE(vURL)
);
```

Example 5 Tuples:

iID	iName	vURL
100	CS: GO.mkv	https://games4fun.com/CSGO.mkv
200	Super Mario.avi	https://games4fun.com/SuperMario.avi
300	NBA 2k18.mp4	https://games4fun.com/NBA2k18.mp4
400	GT Sport. mp4	https://games4fun.com/GTSport. mp4

500	Website Intro.mkv	https://games4fun.com/WebsiteIntro.mkv

WebAdmin (userID: Int, userName: String, password: String, notification: Number, mail: String)

Primary Keys: userID
Candidate Keys: userName
Foreign Keys: None

Functional Dependencies:

userID → {userName, password, notification, mail}
userID is the webadmin ID, the primary key of the table. The primary key implies the rest of the attributes.

userName → {userID, password, notification, mail}
userName is the webadmin name, the candidate key of the table. The candidate key implies the rest of the attributes.

This table is in BCNF because both userID and userName are super key of this table.

SQL DDL

```
CREATE TABLE WebAdmin
(
     userID INT,
     userName CHAR(20)NOT NULL,
     password CHAR(30)NOT NULL,
     notification NUMBER(1,0)NOT NULL,
     mail CHAR(320)NOT NULL,
     PRIMARY KEY (userID),
     UNIQUE (userName)
);
```

Example 5 Tuples:

userID	userName	password	notification	mail
1	admin1	ha	1	111111111@haha.com
2	admin2	haha	0	22222222@haha.com
3	admin3	hahaha	1	33333333@haha.com

4	admin4	hahahaha	0	44444444@haha.com
5	admin5	hahahahaha	1	55555555@haha.com

PersonalUser (userID: Int, userName,password:Char, country: String, gender: String, age:Int, mail: String, Notification: Number)

Primary Keys: userID
Candidate Keys: userName
Foreign Keys: None

Functional Dependencies:

userID → {userName,password,country, gender, age,mail,notification}
userID is the User ID, the primary key of the table. The primary key implies the rest of the attributes.

userName → {userID,password,country, gender, age,mail,notification}
username is a candidate key of the table, it implies the rest of the table as user ID.

This table is in BCNF because userID and userName are super keys of this table.

SQL DDL

```
CREATE TABLE PersonalUser
(
     country CHAR(20),
     gender CHAR(10),
     age INT,
     userID INT,
     userName CHAR(20)NOT NULL,
     password CHAR(30)NOT NULL,
     notification NUMBER(1,0)NOT NULL,
     mail CHAR(320)NOT NULL,
     PRIMARY KEY (userID),
     UNIQUE (userName)
);
```

Example 5 Tuples:

userID	userName	password	country	gender	age	mail	Notification
11	Player_1	12345678	China	Male	20	Player_1@gmail.com	0
12	Player_2	33333333	Canada	Male	38	Player_2@gmail.com	1
13	Exaid	3a3a3a3a	Canada	Female	23	Exaid@gmail.com	0
14	Chain	zzzzz3333	Japan	Female	32	Chain@gmail.com	1
15	Ted	23571113	Canada	Male	19	TTEEDD@gmail.com	1

businessUser (userID: Int, userName: String, password: String, notification: Number, mail: String, officialSite: String)

Primary Keys: userID
Candidate Keys: userName
Foreign Keys: None

Functional Dependencies:

userID → {userName, password, notification, mail, officialSite}
userID is the user ID, the primary key of the table. The primary key implies the rest of the attributes, once we know the business Id, we can know the rest attributes.

userName → {userID, password, notification, mail, officialSite}
This is true because userName is a candidate key, which can also implies the rest attributes.

This table is in BCNF because in both the FDs, userID and userName are super keys of this table.

SQL DDL

```
CREATE TABLE BusinessUser
(
     userID INT,
     userName CHAR(20)NOT NULL,
     password CHAR(30)NOT NULL,
     notification NUMBER(1,0)NOT NULL,
     mail CHAR(320)NOT NULL,
     officialSite CHAR(100),
     PRIMARY KEY (userID),
	UNIQUE (userName)
);
```

Example 5 Tuples:

userID	userName	password	mail	Notification	officialSite
21	CS_GO	12345678	CS_GO@gmail.com	0	csgo.com
22	WOW	33333333	WOW@gmail.com	1	wow.com
23	GT_SPORTS	3a3a3a3a	GT_SPORTS@gmail.com	1	gt.com
24	PUBG	zzzzz3333	PUBG@gmail.com	1	pubg.com
25	NBA_2K	23571113	NBA_2K@gmail.com	0	nba2k.com

ThumbUp (userID: Int, rID: Int)

Primary Keys: {userID, rID}
Candidate Keys: None
Foreign Keys: userID references PersonalUser, rID references Review

Functional Dependencies:

{userID, rID} → {userID, rID}
userID is the Personal User ID, rId is the review ID, and {userID, rID} is the primary key of the table. The primary key implies the rest of the attributes.

This table is in BCNF because {userID, rID} is a super key of this table.

SQL DDL

```
CREATE TABLE ThumbUP
(
	userID INT,
	rID INT,
	PRIMARY KEY (userID, rID),
	FOREIGN KEY (userID) REFERENCES PersonalUser(userID),
	FOREIGN KEY (rID) REFERENCES Review(rID)
);
```

Example 5 Tuples:

userID	rID
11	1
11	3
14	2
15	1
12	4

category (cName: String, totalNum: Int)

Primary Keys: cName
Candidate Keys: None
Foreign Keys: None

Functional Dependencies:

cName → {cName, totalNum}
cName is the name of category, the primary key of the table. The primary key implies the rest of the attributes, once we know the category name we can know total number of game in the category.

This table is in BCNF because cName is a super key of this table.

SQL DDL

```
CREATE Category
(
      cName char(20),
      totalNum INT,
      PRIMARY KEY (cName),
);
```

Example 5 Tuples:

cName	totalNum
free	0
Action	2
RPG	1
Racing	1
Sports	1

game (gameID: Int, gName: String, since: Int, userID: Int)

Primary Keys: gameID
Candidate Keys: {gName, since}
Foreign Keys: userID

Functional Dependencies:

gameID → { gName, since, userID}
gameID is the game ID, the primary key of the table. UserID is the business user ID. The primary key implies the rest of the attributes. Specifically, when we know the gameID, we can know the name, release date and the company of game.

{gName, since} → { gameID, userID}
This is true because {gName, since} is a candidate key. All the games developed by a company cannot have the same release date and game name at the same time.

This table is in BCNF because in both the FDs, gameID and {gName, since} are super keys of this table.

SQL DDL

```
CREATE TABLE Game
(
     gameID INT,
     gName CHAR NOT NULL,
     since NUMBER(4,0)NOT NULL,
	userID INT NOT NULL,
	PRIMARY KEY (gameID),
	FOREIGN KEY (userID) REFERENCE businessUser (userID)
			ON DELETE CASCADE
			ON UPDATE CASCADE
);
```

Example 5 Tuples:

gameID	gName	userID	since
1	NBA 2K18	10	2017
2	PUBG	9	2017
3	CS:GO	6	2012
4	GT Sport	8	2001
5	WOW	7	2002

Belong (gameID: Int, cName: String)

Primary Keys: {gameID, cName}
Candidate Keys: None
Foreign Keys: gameID references Game, cName references Category

Functional Dependencies:

{gameID, cName} → {gameID, cName}
Once we know gameID and cName, we know all the attributes since there are only two attributes.

This table is in BCNF because there is no non-trivial dependency.

SQL DDL

```
CREATE TABLE Belong
(
	gameID INT,
	cName CHAR(20),
      PRIMARY KEY (gameID, cName),
      FOREIGN KEY (gameID) REFERENCES Game(gameID)
            ON DELETE CASCADE
            ON UPDATE CASCADE
      FOREIGN KEY (cName) REFERENCES Category(cName)
            ON DELETE CASCADE
            ON UPDATE CASCADE
);
```

Example 5 Tuples:

gameID	cName
1	Sports
2	Action
3	Action
4	Racing
5	RPG

Contain (rID: Int, iID: Int)

Primary Keys: {rID, iID}
Candidate Keys: None
Foreign Keys: rID references Review, iID references Image

Functional Dependencies:

{rID,iID} → {rID, iID}
we know both of reviewID and imageID then we identify the review contains the image.

This table is in BCNF because there is no non-trivial dependency.

SQL DDL

```
CREATE TABLE Contain
(
      rID INT,
      iID INT,
      PRIMARY KEY (rID, iID),
      FOREIGN KEY (rID) REFERENCES Review(rID)
            ON DELETE CASCADE
            ON UPDATE CASCADE
      FOREIGN KEY (iID) REFERENCES Image(iID     )
            ON DELETE CASCADE
            ON UPDATE CASCADE
);
```

Example 5 Tuples:

rID	iID
1	200
2	300
3	300
4	100
5	400

Has (rID: Int, vID: Int)

Primary Keys: {rID, vID}
Candidate Keys: None
Foreign Keys: rID references Review, vID references Video

Functional Dependencies:

{rID,vID} → {rID, vID}
we know both of review ID and Video ID then we identify the review contains the video.

This table is in BCNF because there is no non-trivial dependency.

SQL DDL

```
CREATE TABLE Has
(
      rID INT,
      vID INT,
      PRIMARY KEY (rID, vID),
      FOREIGN KEY (rID) REFERENCES Review(rID)
            ON DELETE CASCADE
            ON UPDATE CASCADE
      FOREIGN KEY (vID) REFERENCES Video(vID     )
            ON DELETE CASCADE
            ON UPDATE CASCADE
);
```

Example 5 Tuples:

rID	vID
2	200
3	100
4	200
2	100
5	500