CSIT115 Data Management and Security
Laboratory 2

Scope

This laboratory includes the following:

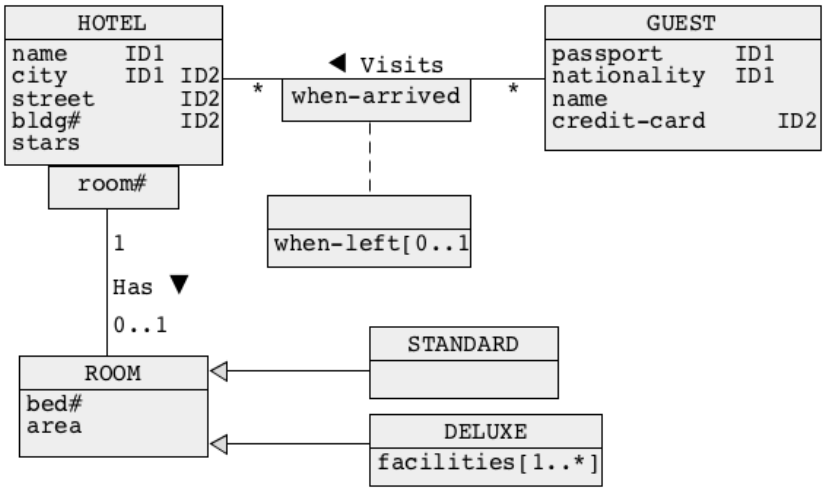
- tasks related to logical modelling and relational data model
- tasks related to quality evaluation of relational database design and `CREATE TABLE` statement of SQL

Specification of each task starts from a new page.

It is recommended to solve the problems before attending an enrolled laboratory class in order to efficiently use supervised laboratory time.

Task 1

Consider a conceptual schema given below.



Your task is to perform a step of logical database design, i.e. to transform a conceptual schema given above into a collection of relational schemas.

For each relational schema created clearly list the names of attributes, primary key, candidate keys (if any), and foreign keys (if any). Assume, that **subset method** must be used to implement a generalization (if any). A way how a conceptual schema can be transformed into a collection of relational schemas is explained in a presentation 06 Logical Design.

The relational schemas must be listed in a format presented in the last few slides in a presentation 06 Logical Design.

Task 2

Consider the following collection of relational schemas.

```
PROPERTY(PNum, City, Street , HouseNum, OwnerPhone, Price)
Primary key = (PNum)
Candidate key = (City, Street, HouseNum)
Foreign key = (OwnerPhone) references OWNER(OwnerPhone)

OWNER(OwnerPhone, OwnerName)
Primary key = (OwnerPhone)

BUYER(BuyerPhone, BuyerName, City, Street, HouseNum)
Primary key = (BuyerPhone)
Candidate key = (City, Street, HouseNum)

PREFERENCE(BuyerPhone, City, Street, MaxPrice, MinPrice, PDate)
Primary key = (BuyerPhone, PDate)
Foreign key = (BuyerPhone) references BUYER(BuyerPhone)

INSPECTION(BuyerPhone, PNum, IDate)
Primary key = (BuyerPhone, PNum,  IDate)
Foreign key 1 = (BuyerPhone) references BUYER(BuyerPhone)
Foreign key 2 = (PNum) references PROPERTY(PNum)
```

Your task is to perform *reverse database engineering*, i.e. to find a conceptual schema of a database that has a collection of relational schemas given above. Use UMLetlet to draw a conceptual schema found. Use an option File->Export as... to export your diagram into a file `solution2.bmp` in `BMP` format.

Task 3

Analyze the relational schemas listed below and find the schemas that are incorrectly designed. A schema is incorrectly designed when the insertions of rows into the schema creates the redundancies. Consider a method of row insertions explained in a presentation 07 Database Design Quality to find out which relational schemas are incorrectly designed.

Next, for each incorrectly designed relational schema propose its decomposition into a pair of correctly designed relational schemas and equivalent to the original one. A pair of relational schemas is equivalent to other relational schemas when the respective pair of relational tables contains exactly the same information as a relational table built on the original schema.

(1) `PLAYER(pnum, pname, team, position, team-address)`
A relational schema `PLAYER` is a header of a relational table that contains information about the players and teams the players belong to. A player number (`pnum`) uniquely identifies each player in a team; and team name (`team`) uniquely identifies the team address (`team-address`). The attribute `position` describes the player's position in a team.

(2) `WAREHOUSE(name, address, item, price, quantity )`
A relational schema `WAREHOUSE` is a header of a relational table that contains information about the warehouses and items kept there. An attribute `address` is an address of a warehouse and it uniquely identifies each warehouse. An attribute `name` is name of a warehouse and it also uniquely identifies each warehouse. An attribute `item` is a unique name of an item. A pair of attributes (`address`, `item`) uniquely identify the item's unit price (attribute `price`) and quantity (attribute `quantity`) of the item.

(3) `FLIGHT(passenger, flightNum, seatType, departurePlace, departureTime)`
A relational schema `FLIGHT` is a header of a relational table that contains information about the flights booked by the passengers. A passenger is uniquely identified by a value of an attribute `passenger`. A passenger books seats (`seatType`) on many different flights (`flightNum`). Only one flight (`flightNum`) takes off from a given departure place (`departurePlace`) at a given departure time (`departureTime`). A flight number (`flightNum`) uniquely determines departure place (`departurePlace`) and departure time (`departureTime`).

(4) `ENROLMENT(studentNum, subjectCode, enrolDate, IP)`
A relational schema `ENROLMENT` is a header of a relational table that contains information about the students enrolled in the subjects (`subjectCode`). A triple of attributes (`studentNum`, `subjectCode`, `enrolDate`) uniquely identifies each row in the relational table. An attribute `IP` is an IP address of a computer from where an enrolment has been done.

(5) `CAR(regoNum, manufacturer, model, year, colour)`
A relational table `CAR` contains the descriptions of cars. A registration number (`regoNum`) uniquely identifies each car. A car is described by a manufacturer name (`manufacturer`), model name (`model`), year when manufactured (`year`), and colour (`colour`).

No new attributes can be added to the relational schemas listed above !

The relational schemas must be listed in a format presented in the slides 44-45 in a presentation 06 Logical Design.

For each one of the cases from (1) to (5) write either a statement:

A relational schema <schema-name> is designed correctly

or a statement:

A relational schema <schema-name> is designed incorrectly

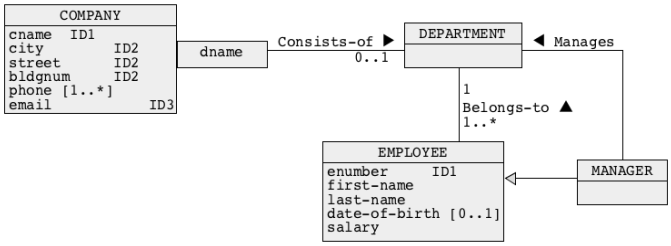
<schema-name> must be replaced with a name of a respective relational schema.

For each incorrectly designed schema, show the results from an insertion test that reveal the redundancies in a relational table that has incorrectly designed relational schema. Insert precisely 3 rows into a relational table.

Additionally, for each incorrectly designed relational schema list a correctly designed relational schemas obtained from a decomposition of the incorrect schema. The relational schemas must be listed in a format presented in the last few slides in a presentation 06 Logical Design.

Task 4

Consider a conceptual schema given below.



Implement SQL script `solution4.sql` with `CREATE TABLE` statements that implement the relational schemas obtained from transformation of the conceptual schema given above into the relational schemas. Use a superset method for implementation of generalization.

Note, that you MUST use only `CREATE TABLE` statements and no other statement of SQL !

You can find a lot of information about implementation of `CREATE TABLE` statements in a presentation 09 SQL - Data Definition Language (DDL) and in Cookbook, How to use data definition and basic data manipulation statements of SQL, Recipe 4.1 How to create and how to alter the relational tables?

When a script file `solution4.sql` is ready connect to MySQL either through command line interface mysql or graphical user interface MySQL Workbench and execute your script file.

If processing of the file returns **any errors** then you must eliminate the errors! Processing of your script must return **NO ERRORS** !

It is recommended to create a script `dbdrop4.sql` that drops all relational tables created during execution of a script `solution4.sql` and it is recommend to execute `drop4.sql` after each execution of `solution4.sql` (you can also put `DROP TABLE` statements at the end of a script `solution4.sql`). In such a way you can avoid an unpleasant syntax error messages like:

```
ERROR 1050 (42S01): Table '…' already exists
```

when you execute a script `solution4.sql` the next time. **Please, remember that such message also count as an error in processing of the script with all consequences coming from such fact.**

When execution of your script returns no errors connect to MySQL server using command based interface mysql and create a report from processing of the script `solution4.sql`. Save your report in a file `solution4.rpt`. To create a report you must use a command `tee solution4.rpt` before processing the script and a

command `notee` after processing of a script. Then, you can find a file `solution4.rpt` in the current folder of mysql client. You can find more information about creating reports from processing of SQL scripts in Cookbook, Recipe 3.1 How to use "mysql? Command based interface to MySQL database server? Step 4 How to save the results of SQL processing in a file?

Your report must contain a listing of SQL statements processed. To achieve that, you must logon mysql client with **`-v`** (`verbose`) option in the following way:

```
mysql -u csit115 -p -v
```

You can find more information on how to display SQL statements while a script is processed in Cookbook, Recipe 3.1 How to use "mysql? Command based interface to MySQL database server ? Step 3 How to process SQL script ?.

You must use mysql client to create a report !

And again, … a report from processing of SQL script must contain NO ERRORS !