THE UNIVERSITY OF SYDNEY

School of Information Technologies
A/Prof. Uwe Roehm

INFO2120: Database Systems I **Semester 1, 2015**

Assignment 1 Brief: Conceptual Database Design

1 Introduction

This assignment is about the conceptual design for a database. The objectives are to gain practical experience in conceptual database modelling based on a requirements document and to experience group-based data design work. This is a group assignment for teams of about 3 members. You should inform your tutor of your group's composition by the Week 3 tutorial. Please also keep an eye on the discussion forum and further announcements in Piazza.

2 Submission Details

The final version of your database model should be submitted electronically via eLearning by midnight (end-of-day) Monday Week 5.

2.1 Submission Items

Please submit an Entity-Relationship diagram following the conventions presented in the Conceptual Database Design lecture material from Week 2. Your diagram should ideally be in PDF format, but an image format such as PNG, GIF or JPEG is also acceptable. You can annotate your diagram with comments where necessary, in particular clearly document all design extensions. The target paper size should be A4. If required you can spread your diagram over several pages.

2.2 Academic Honesty / Plagiarism

By uploading your submission to eLearning your group implicitly agrees to abide by the University policies regarding academic honesty, and in particular that all the work is original and not plagiarised from the work of others. If you believe that part of your submission is not the work of your group members you must bring this to the attention of your tutor or lecturer immediately. See the policy slides released in Week 1 for further details.

In assessing a piece of submitted work, the School of IT may reproduce it entirely, may provide a copy to another member of faculty, and/or communicate a copy of this assignment to a plagiarism checking service or in-house computer program. A copy of the assignment may be maintained by the service or the School of IT for the purpose of future plagiarism checking.

2.3 Late submissions

Please start early so that your tutor can give you feedback on your approach in the tutorials of Week 3 and 4. The final submissions deadline is Monday of Week 5. An example solution to the assignment will be presented in the week of the final submission deadline, so please keep to the deadline. Late submissions will be penalised 20% per day late.

3 Marking

This assignment is worth 10% of your final grade for this unit of study. Your group's final submissions will be marked for correctness and completeness of the database design according to the following rubric. Please note that the mark awarded for your assignment is conditional on *every team member* being able to explain details of your designs to your tutor or the lecturer if asked.

3.1 Rubric

Your submissions will be marked according to the following rubric, with a maximum possible score of 10 points.

	Novice (0 pts)	**Competent (1 pt)**	**Proficient (2 pts)**
Notation	Bigger mistakes in the usage of E-R notation	Good usage of E-R notation with a few mistakes	Proficient usage of the E-R notation
Core Model	Less than competent model of the given scenario	Not all major entities and relationships of core model are correctly captured	The core model was very well designed in the model
Constraints	No constraints captured at all	Some constraints (multiplicities and cardinality constraints on relationships, etc.) are included in the model, but either incorrectly or incomplete	All given integrity constraints are modelled correctly
Design Specialities	No design specialities used	At least one useful ISA, weak entity or aggregation used in a meaningful way	All appropriate design specialities are used appropriately
Extension(s)	No extensions beyond the core design brief	One extension was attempted, but either with mistakes or not very substantial, or the textual description/explanation was missing	At least one substantial extension to the given scenario was included, described and modelled correctly

3.2 Feedback

If you present a draft of your model in the tutorial prior to the submission deadline to your tutor, you will receive feedback with suggestions on key areas for improvement. Your group's final submission will receive general feedback, explaining your final mark with respect to the rubric.

3.3 Group Member Participation

If members of your group do not contribute sufficiently you should alert your tutor as soon as possible. The tutor has the discretion to scale the group's mark for each member as follows:

Level of contribution	Proportion of final grade received
No participation	0%
Full understanding of the submitted work	50%
Minor contributor to the group's submission	75%
Major contributor to the group's submission	100%

3.4 General Guidelines

- Your diagram should be clearly laid out and easy to interpret.
- Take care to consider how individual entities and relationships should be uniquely identified, and **avoid introducing artificial keys** beyond those as specified in the brief.
- Remember that an entity type's identifiers (keys) should not appear as attributes of other entity or relationship types.
- Remember that you have **aggregation** and **weak entities** available to deal with cases where a full key is not possible, but avoid gratuitous use of them except where necessary.
- You should also look for an opportunity to demonstrate your understanding of **IsA hierarchies**, but again be prudent.
- Don't forget to consider what **constraints** should apply to the participants of any relationships as these are crucial for correctly mapping to a logical database design (as you will see in Assignment 2).

4 Design Brief: PeerPark Parking Space Sharing

It is difficult to find a place to park your car in many cities, even though many parking spots go unused. PeerPark aims to revolutionise commuter life by connecting the owners of parking spaces with the drivers who need them, in a similar spirit to companies like GoGet (car sharing), Uber (ride sharing) and AirBnB (room sharing). PeerPark is a startup established by engineers who have developed two low-cost technologies to make this revolution possible:

ParkTag
: This small unit is attached to the windscreen of a car and allows the car to be identified when it parks in a bay.

ParkPod
: This robust device attaches to the ground in the middle of the parking space and monitors what parks over it, notifying our central server.

Initial small-scale trials of PeerPark have been very promising, but data is currently managed in an ad-hoc manner. PeerPark's founders realise that their data needs to be properly handled in a central database, and have asked you to design one for them. You should use the following design description to create a conceptual model for the data that must be stored in this database.

4.1 Core Database

Each car registered with PeerPark needs to have its registration number, make and model recorded. Because some parking bays have size restrictions we also need to record the car's dimensions (width, length and height). Every car is issued with a ParkTag device with a unique device ID (a 10-digit alphanumerical code). ParkTags are never re-used for other cars, but a car may have several tags issued to it (to replace damaged tags, for instance), but never simultaneously.

Details of every rentable parking bay must be recorded, including its address and GPS position (longitude and latitude) and some optional description. In some cases there may be several bays at the same address (for instance, at a block of units) so an extra site detail also needs to be captured (e.g., 'Carport', 'Bay 2', etc.). Each bay has its own ParkPod device, allowing us to track what gets parked at the bay (for billing purposes, and to alert the owner if an unregistered vehicle is parked there). Each ParkPod has its own device ID of the same format as the ParkTag devices. For communication purposes each ParkPod also has a mobile phone SIM fitted, so we need store the device's phone number. We also need to know the size of each bay to check that a car will fit.

We need to store some general information about the system's members, such as their full name, birthdate, gender (or title) and address. We also need at least one phone number by which we can contact them. In order to be able to use our system via the website, members can log in using their email address along with a password. We also keep billing information per member, such as either a bank account, PayPal or credit card (with corresponding details). A member can have up to three different billing accounts, with one preferred account.

Members can be drivers or parking-bay-owners, or both. For drivers we need to know what cars they own, and for legal reasons we need to know the driver's license number, and until when it is valid. Drivers may nominate favourite parking bays to facilitate faster bookings, rather than searching each time. Drivers can give their cars names, to tell them cars apart easily. For bay owners we need to know what bays they own. A member can specify available periods for which a bay is available. This is specified as a week-day period (begin-end) and weekend period.

A chief function of the database is to track bookings. Members can make bookings for bays for a certain time period for one of their cars. The booking system must keep track of when bookings have been made by members, for which exact time, and for how long. For simplicity reasons, we only allow bookings for a full hour, starting at a full hour.

4.2 Extensions

The following are areas in which the database can be extended to add extra functionality. They are not clearly defined, so you will need to propose reasonable design considerations for your model. The extensions are also of varying difficulty. You should include at least one extension to get full grades for your submission. You can develop an alternative extension, provided you get agreement from your tutor. Remember that these extension should provide a significant addition to the core model, and are opportunities to demonstrate your ability to use design specialities not required in the core model.

4.2.1 Ratings and Reviews

We would like to be able to incorporate members' opinions of the available parking bays. Members should be able to provide a rating from 1 (very poor) to 5 (very good) for any parking bays that

they have used. They can also include a comment explaining their rating. Other members shall further be able like or dislike ratings based on how useful they found them. For each rating, review or like/dislike, the database shall keep track of who issued those ratings and when . We would also like to allow users to give themselves a nicknames that can be shown in their reviews (to avoid showing email addresses). Along with a nickname, we should also record as some statistics on each member, such as how many bookings they have made themselves, how long they have been a member and how many reviews they have written.

4.2.2 Logging

Our ParkPod devices periodically transmit log data to our central server, and we would like to capture this data within the database . We would like to record when a log was received from a device, and which bay it is for (devices are sometimes moved between bays, e.g., for maintenance). We also want to store the actual log data. Each line in the log has the following details, describing each time the bay was occupied:

- Start Timestamp when bay is occupied
- End Timestamp when bay is departed
- ParkTag ID of the vehicle occupying the bay, if known

This data can be used for invoicing members, with fines applied when bays are occupied outside of their booked periods. Note that a user can occupy their car in a bay multiple times during the period that s/he has booked it.

4.2.3 Plans and Invoicing

We would like to offer a variety of billing plans, both for drivers and bay owners. In both cases we would offer a monthly fee. At present we charge a set hourly fee for renting any bay, or a fixed daily fee for anything over 12 hours. If we offered different plans we might set lower rental fees for a higher monthly fee. We would also be interested in being able to allow premium bay owners to set per-bay fees.

Members should receive monthly activity statements, and the details of these should be stored in the database. Each member's activity statements have an identifying number (starting at one and incrementing for each statement for that member). The activity statements should capture all bookings made by the member, along with the cost they will be debited; and all bookings made of the member's bays, along with the cost they will be credited. The statement should also include the total cost of the activity, including the monthly fee.