**Part 1: Database Concepts**

**Subpart A**

According to Coronel & Morris (2015) structural independence “exists when you can change the file structure without affecting the applications ability to access the data ” (p. 19). When the structural independence of the Innkeepers IT system is analyzed we find that modifications to the data stored within the XML files could lead to additional changes in other files. For example, the names and locations are currently being stored in one XML file while the room reservations and booked guests are stored in other XML files that get generated daily. If the hotel decides it wants additional data to be stored on their guests such as arrival time and checkout time multiple files would have to be updated due to their interdependence on one another. If the hotel decided to track the cleaning and maintenance of individual rooms during the times when guests were not occupying them it would need to adjust the underlying structure of its two existing XML files which in turn could lead to errors.

The ElectrInn system which utilizes a database would allow the hotel to get away from the structural dependence of the current XML file system and move towards structural independence. The ElectrInn application could easily store data about guests in multiple tables, and updating one table would not cause another table to be changed. For example, there could be a “Booked\_guest” table with data about each guest such as name, duration of stay, check-in and check-out times. A “Room” table that holds data about the characteristics of a hotel room such as room number, occupancy limit, and number of beds. A “Reservation” table could hold data on the dates guests have reserved for their stays and the proposed length of their stays.

Since the ElectrInn application makes use of a database if the hotel were to now decide it wanted to store additional columns in the “Booked\_guest” table those fields could be easily added without any changes to the “Room” or “Reservation” tables. The tables would ideally be linked by using a foreign key. For example, the “Room” table could have a foreign key called “guest\_id” which would also be present in the “Booked\_guest” table. This way additional columns could be added to the “Booked\_guest” table and the “Room” table which would have a reference to it would not need to be changed and would still be able to show what guest was booked into a certain room.

In regards to data redundancy since both the Innkeepers and ElectrInn applications allow for users to connect to central repositories where the information is stored there should theoretically be no “islands of information” where different locations store the same content (Coronel & Morris 2015, p. 20). However, with the Innkeepers application which relies on XML files saved on a file server there is temptation for a developer to create redundant data across several different XML files rather than to try modifying existing XML files which could potentially lead the developer down a path were many more XML files end up needing to be updated due to the underlying structural dependence of the Innkeepers system.

Data redundancy causes a multitude of problems such as “poor data security”, “data inconsistency”, “data-entry errors”, and “data integrity problems” (Coronel & Morris 2015, p. 20). A security issue could potentially be created in the Innkeepers application if the same content was to be spread across multiple XML files. A file server offers some level of protection against unauthorized access attempts by enforcing access controls on the directories where the XML files reside and on the XML files themselves, but it certainly does not offer the granular permission levels that a DBMS such as Electrinn can offer. A database can be restricted to provide row level access to certain users, as well as to be encrypted.

Data inconsistency and data-entry errors can also occur with data redundancy which can in turn lead to users not trusting the data or the overall application that houses it. Suppose both the XML files for booked guests and reservations contain a field called “guest\_name” and in one file the guest is listed as “John Smith” by one user and in the other file they are listed as “Jon smith” (without the “h”) by another user. It would be difficult to tell if it was the same guest or two different guests with the same name but spelled differently.

Data redundancy can lead to security issues, data inconsistency, or entry errors which cause users to not be able to make sense of, or even trust their own data. When data can no longer be trusted to be true and accurate there is an integrity issue. Data integrity is important since it does not make sense to collect data in the first place if it cannot be turned into information which can then be used for decision making.

When it comes to analyzing system complexity the Innkeepers XML based IT system may suffice when there are small amounts of data. However if the hotel were to renovate each year and add more rooms it would be doubtful that the Innkeepers system could scale and continue serving the hotels needs. It would be more efficient to have the ElectrInn system scale its database out as the hotels needs changed. Also the maintenance of the database based application would be much easier for tasks such as backing up data or restoring accidentally deleted content. Having the less complex ElectrInn database system could potentially be less costly in the long run when maintenance and development costs are factored into the equation. A more complex system will always eat up more of the developers and administrators time which can lead to cost overruns.

**Subpart B**

The way users interact with the hardware, software, procedures, and data of the countryside hotel could all potentially be altered depending on which one of the two IT systems they end up using.

When it comes to hardware both the Innkeepers and ElectrInn applications should both be hosted on servers. There may be some differences in server specifications if the hotel was to switch over to the ElectrInn system due to the fact that a RDBMS would have certain minimum hardware specifications such as RAM and CPU cores that may need to get upgraded before the RDBMS could be installed. From a user standpoint there shouldn’t be much of a change due to how the applications are hosted. Also the hotel employees will continue to use the same desktop and laptop computers that they are currently using regardless of what system they use to keep track of the guests.

One of the major areas where users would see a difference is in how they interact with the software that they use to manage the hotel. The underlying operating system software of the desktop and laptop computers should stay the same, however, the software that the users interact with to enter information about guests, rooms, and reservations will change. The interfaces for the Innkeepers and ElectrInn systems will most likely be vastly different so the hotel employees would have to get trained on how to use the new software.

The hotel would have to hire a database administrator or would have to train one of the existing IT staff members if they were to go with ElectrInn since they now have a DBMS to support. They may also need to hire a database developer if they want to customize the application or it’s reporting features. This would however allow for the current hotel employees to access their data in a completely new way by having enhanced reporting capabilities, improved security, backup and recovery options for their data.

Certain procedures for users would be different between the ElectrInn and Innkeepers applications. The way that users perform some aspects of their jobs could be slightly altered depending on what IT system was used by the hotel. For example, it would be much easier to monitor the data using a DBMS than XML files on a file server. It would also be easier to generate ad hoc reports from the data which could then be used to drive decisions, as a result business procedures may change due to the increased access to available information. Also, Electrinn which uses a DBMS would allow for concurrent connections to the data, so procedures for the Innkeepers application where only one user at a time could update anything due to it using an XML file would change. This would result in increased productivity for users at the hotel.

The data stored within the two systems would be identical in nature, however, with the caveat that with a DBMS it would be much easier to expand the amount of data stored as well as the structure in how it is stored. A DBMS would also allow for better reporting and analysis of the data, and could help prevent security compromises by safeguarding the data either through access control policies or encryption.

**Subpart C**

There are certain advantages as well as disadvantages for the countryside hotel to switch from the Innkeepers to the ElectrInn IT system. For instance, a drawback to updating the IT system is the fact that the staff of the hotel would have to be retrained in certain aspects of their jobs. Also, it could potentially cost the hotel more money up front to switch to ElectrInn because they would have to bring on additional personnel to manage, customize, and maintain the DBMS.

On the other hand upgrading to a DBMS, although it may bear some upfront costs could potentially help the hotel save money in the long run by allowing for better, faster, and more secure access to data that can be turned into information which drives decision making for the organization. The ElectrInn system would also allow for data integrity because data would be stored in database tables as opposed to numerous XML files. The data could be accessed concurrently by multiple hotel employees so productivity should rise. Reporting and ad hoc queries of the data would also significantly help to streamline the hotels business capabilities.

The benefits gained from switching over to the ElectrInn system which comprises a DBMS far outweigh any downside risks. The countryside hotel simply cannot scale it’s operations or efficiently utilize its own data to drive decision making by relying on XML files stored on a file server. A DBMS such as ElectrInn must be incorporated into the hotel in order for the organization to probably analyze and take advantage of its own data and turn it into useful and relevant information.

# **Reference list** Coronel & Morris (2015). Database systems: Design, implementation, and management. Boston, MA: Cengage Learning.

**Part 2: Converting a Spreadsheet to a Database**You might encounter something akin to the table below in a spreadsheet or file-based application.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Make** | **Model** | **ID** | **Miles Driven** | **Date Driven** | **Mileage Cost** | **Driver ID** | **License Number** |
| Honda | Accord | 10 | 38.5 | 10/23/2012 | $20.60 | 101 | 032 592 173 |
| Honda | Accord | 10 | 38.5 | 1/17/2012 | $20.60 | 101 | 032 592 173 |
| Jaguar | I-PACE Concept | 20 | 90 | 3/1/2012 | $48.15 | 119 | 331 997 012 |
| Jaguar | I-PACE Concept | 20 | 90 | 2/22/2012 | $48.15 | 237 | 419 953 551 |
| Jaguar | I-PACE Concept | 209 | 20 | 3/6/2012 | $10.70 | 119 | 331 997 012 |
| Porsche | 718 | 30 | 18 | 10/23/2012 | $9.63 | 119 | 331 997 012 |
| Lamborghini | Roadster | 40 | 40 | 1/17/2012 | $21.40 | 213 | 945 459 123 |
| Toyota | Camry | 50 | 35 | 1/17/2012 | $18.73 | 314 | 123 476 991 |
| Kia | Optima | 60 | 40 | 1/17/2012 | $21.40 | 101 | 032 592 173 |
| Kia | Optima | 60 | 40 | 1/24/2012 | $21.40 | 149 | 842 248 842 |

Imagine that you want to convert this to a relational database. Answer the following questions, which are progressively more difficult:

1. How many records does this file contain? **10 records.**
2. How many fields are there per record? **8 fields per record.**
3. What are the field names? **Make, Model, ID, Miles Driven, Date Driven, Mileage Cost, Driver ID, License Number.**
4. What do the records in this table probably represent? **The records in this table probably represent drivers and how much money the company is spending on mileage per driver.**
5. Is there data redundancy in the table? If so, describe it. **Yes, there is data redundancy in this table. For instance, the first and second records are essentially the same with the exception of the Date Driven field value being different. When multiple records exist with similar data it could lead to data anomalies where a car model is spelled slightly differently in the numerous records. Also, the database table will be larger than it necessarily has to be since it’s storing the redundant data. The table seems to be designed to hold numerous columns that represent the same data. For example, there is both an ID column and a Model column that seem to be tracking the same car. It would make sense to get rid of the ID column and keep the Model column this way you don’t have an arbitrary ID which is supposed to represent a car Model you would just have the actual car Model itself. Also, why have both a Driver ID and a License Number column? Either column could be used to represent a driver, so one of them could be removed. It may make sense to remove the arbitrary Driver ID assigned by the company and keep the actual license number to track who the drivers are.**
6. Does this table contain data that is part of different entities? If your answer is “yes”, identify and briefly describe the entities. **Yes, there are multiple different entities within this table such as a car and a driver. The car entity has Make, ID, Model, Miles Driven, Date Driven, and Mileage Cost. Then there is an entity for the Driver which has the Driver ID and License columns.**
7. Are there any fields you would add when converting this spreadsheet to a database? If your answer is “yes”, identify each field then briefly describe the reason why you would choose to add it. **Yes, I would add a field, I would split this single table into 2 distinct tables called Car and Driver. The driver table would have License Number and Driver ID and I would add ‘First Name’ and ‘Last Name’ fields to the driver table. Then the remaining columns (Make, ID, Model, Miles Driven, Date Driven, and Mileage Cost) would be in the Car table. In the Car table I would add a foreign key for the drivers ‘License Number’ so that the car is linked with a driver.**
8. Are there other problems you would you need to address in converting this data to a relational database? If your answer is “yes”, identify and briefly describe each problem. **Yes, the database would need to be designed so that the relevant entities were split apart into different tables. A separate Car and Driver table should be created from the spreadsheet. Also, as previously mentioned the redundant data columns should be removed.**