Shadow Paging

It is inconvienient to maintain logs of all transactions fro the purposes of recovery. An alternative is to use a system of shadow paging. This is where the database is divided into pages that may be stored in any order on the disk. In order to identify the location of any given page, we use something called a page table. See the diagram below:

Get image from back of handout with page tables Diagram 8.1

During the life of a tranasacation two page tables are maintained, one called a shadow page table and current page table. When a tranasaction begins both of these page tables point to the same locations (are identical). During the lifetime of a transaction the shadow page table doesn't change at all. However during the lifetime of a transaction changes may be made update values etc. So whenever we update a page in the database we always write the updated page to a new location. This means that when we then update our current page table to reflect the changes that have been made.

Looking at diagram 8.1 we see how these tables appear during a transaction. As we can see the shadow page table shows the state of the database just prior to a transaction, and the current page table shows the state of the database during or after a transaction has been completed.

We now have a system whereby if we ever want to undo the actions of a transaction all we have to do is to do is recover the shadow page table to be the current page table. As such this makes the shadow page table particularly important, and so it must always be stored on stable storage. On disk we store a single pointer location that points to the address of the shadow page table. This means that to swapt the shadow table for the current page table (commiting the data) we just need to update this single pointer (very unlikley to fail during this very short fast operation).

Shadow paging sumary: 1. Modifications made by transactions and still in the buffers are forced to stable DB. 2. Current page table is output to disk 3. Update the disk addresss in the special location.

Disadvantages of the shadow paging system

- Data will suffer from fragmentation as the data is divided up into pages that may or not be in linear order for large sets of related data.
- Garbage will accumulate in the pages on the disk as data is updated and pages loose any references. For example if i have a page that contains a data item X that is replaced with a new value then a new page will be created (remember we always create a new page to update data in a new location). Once the shadow page table is updated nothing will refrence the old value of X.

Recovering from Media failure

In this situation there is very little that you can do. The best code of practice is to make **regular** backups of your data and store these backups in a safe and physically different location to the main database. If you take a log this should also be backed up, and this combined with the most recent database archieve should help you recover the most up-to-date and consistent database as possible

Data Modelling

In order to correctly create a data model, you must have a good understanding of the following item:

- What the data you are storing is and describes.
- The relationships between the data items.
- Real world constraints. Understanding and identifiying these items is known as 'Semantic Modeling'. One method for semantic modeling is called the entity relationship model (developed by P.P.Chen).

In entity relationship modeling there are three main concepts:

- Entity sets - an entity is a 'thing' or 'object' in the real world that distinguishes from other objects. For example John Smith - National Insurance #: 986654221 (unique to John Smith), or an account number at natwest like: 1246646.
- Attributes - for each entity you must store a set of attributes that fully defines them. For example:

Entity - Customer

Customer name	Customer address	Cust Tel#
Jones	Hurley Hill	1233255234
Smith	Hurley Hill	2340982356

(note there can be duplicate attributes)

Each attribute may have a certain 'domain' which is the range of values they can take. There are two types of attributes, 1. Composite attributes for example customer-name, 2. Simple attributes for example first-name, middle-name, last-name. Attributes can also be either single-valued or multi-valued. Single valued attributes are those that each entity can only take one value of e.g. an employee can only have one name, multi-valued attributes can have multiple values per entity, for example departments an emplyoee is in. Dervived attributes - attributes that can be calculated for example if you know the date an employee started at a company and the current date you can work out how long they have worked there. Null attributes values that are optional per entity. For example model of car owned - not every employee owns a car.

- Relationships