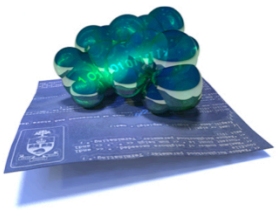
A
Bioinformatics
Course

Databases



Boris Steipe

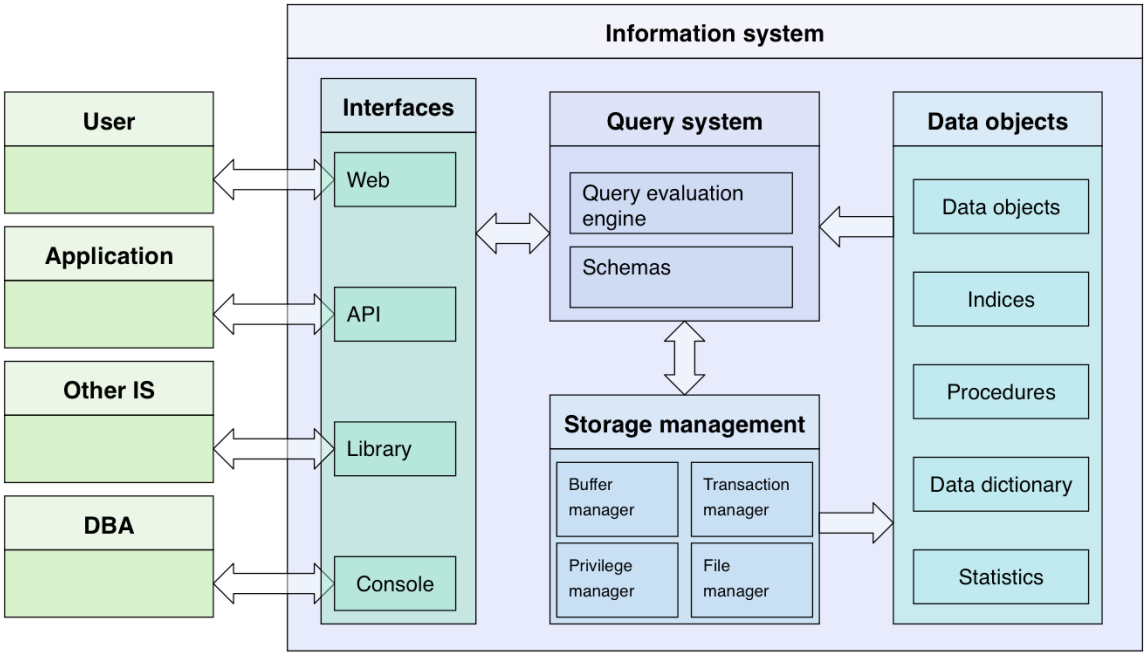
department of biochemistry – department of molecular genetics
University of Toronto

There are many ways to store data and it is important not to be dogmatic about which database system to use. Define your objectives, evaluate alternatives, and make an informed decision.

What do you want to do with the data?

DATABASES

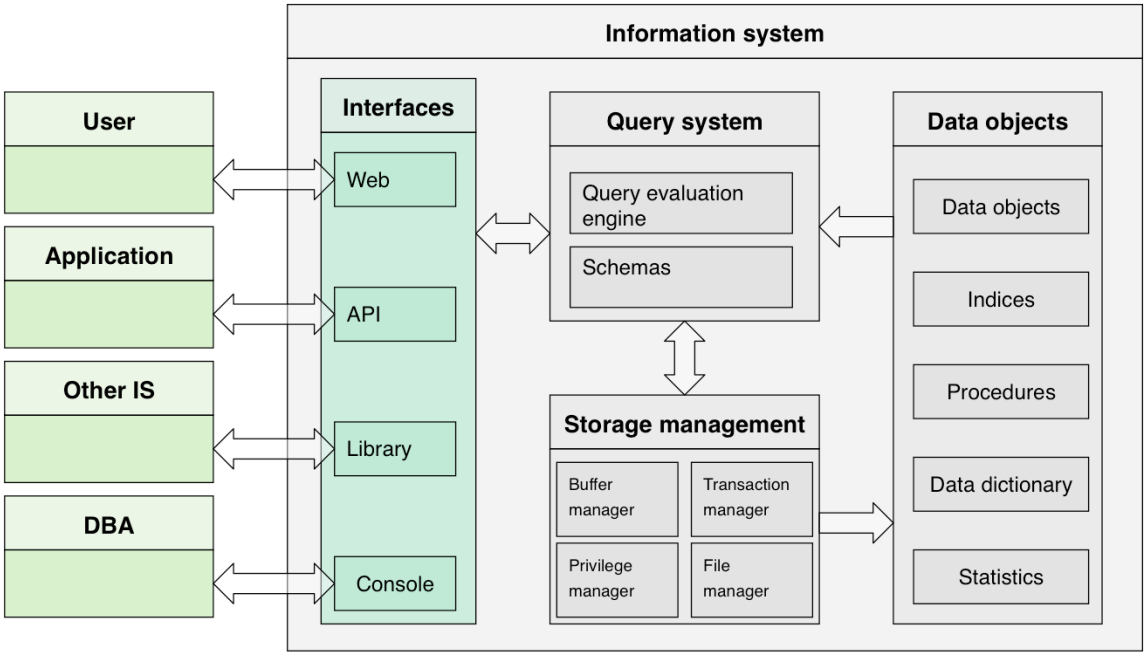
"Database" can refer to the actual data stored, but more generally a "Database" is a layered system of components to store, update and retrieve information.



A "Database system" is much more than just the data it contains. It needs to interface with a variety of different user categories, provide consistent interfaces for querying and retrieval, evaluate queries with respect to the schemas (data models) of the data it hosts, manage the integrity of transactions, and finally, store the actual data and any metadata that is required to describe it.

In our learning units' R code these functions are implemented in an absolutely minimal fashion.

"Database" can refer to the actual data stored, but more generally a "Database" is a layered system of components to store, update and retrieve information.



Four main categories of users are commonly supported by large, public database systems and each has specific interface requirements that correspond to their typical use cases:

Normal **Users** are provided with a simple interface, usually a Graphical Unser Interface (GUI), often realized with forms in Web pages that are dynamically generated, depending on the underlying data. This is the kind of interface you see when you visit NCBI or EBI pages.

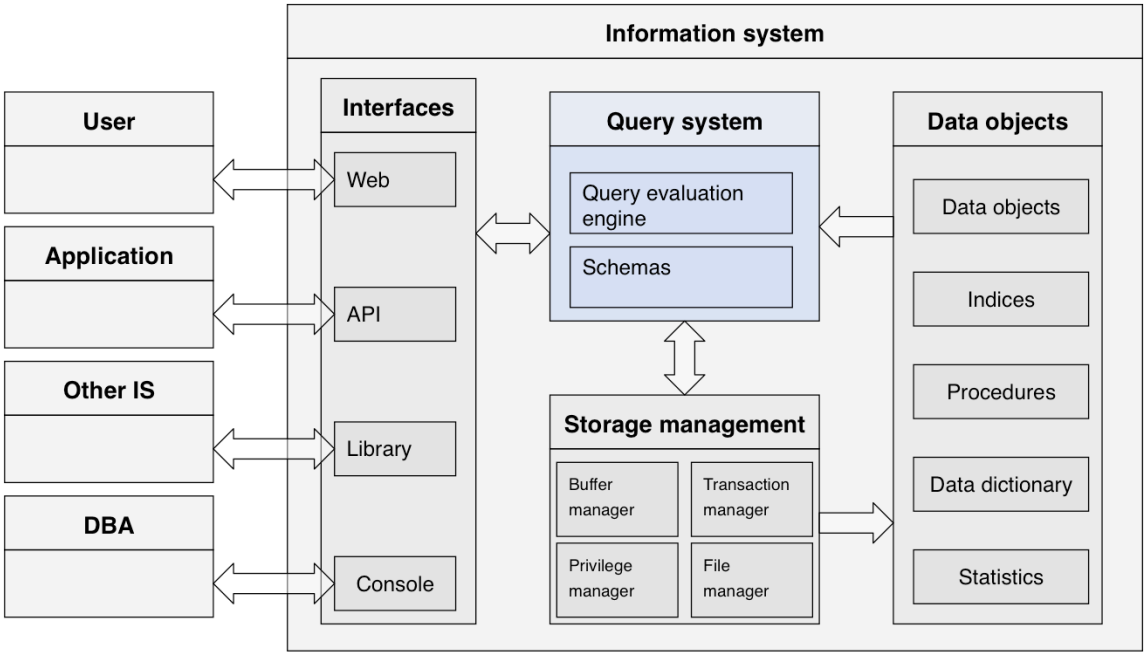
Applications need high-throughput interfaces that support automated, multiple queries. Interaction via mouseclicks or manual entry of data into forms are not suitable. That said, if the database system you are working with does not provide an application interface, you need to simulate user interactions in whatever way is necessary. Or: contact the authors and ask for permission to download the data or provide a useable interface.

Often the application that accesses a database is another database, or **Information system (IS)**. In this case, care must be taken to provide stable accession keys for crossreferencing. Moreover, responsible providers will make code libraries available that can be used to support database access. In R, such libraries might be maintained in R packages.

Finally, the **DBA** (DataBase Administrator) usually has a special interface – often a command line interface – through which she manages user privileges, installs new schemata, deploys stored procedures, tweaks performance etc.

In our R code, we do not distinguish between user roles and all access is via R scripts or R commandline statements.

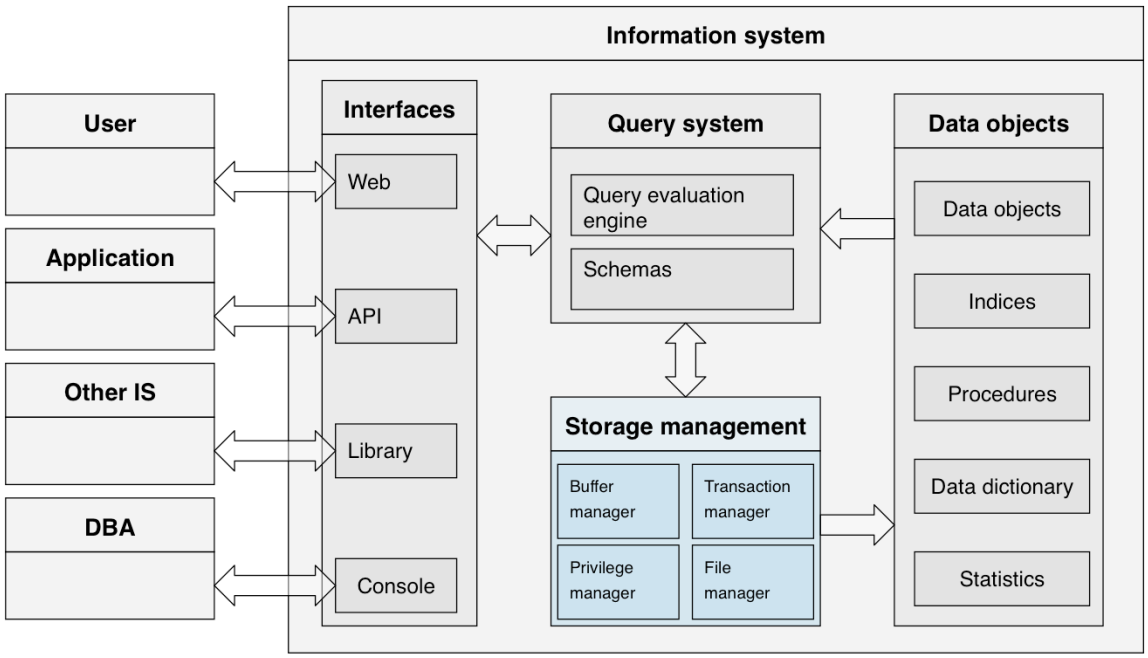
"Database" can refer to the actual data stored, but more generally a "Database" is a layered system of components to store, update and retrieve information.



The **query system** is the brain of the database. It evaluates all interaction requests, translates them into correct and consistent commands for the actual storage engine, and does so in the context of the schema or schemata that describe the actual data.

In our learning units database, queries are done via subsetting of data frames.

"Database" can refer to the actual data stored, but more generally a "Database" is a layered system of components to store, update and retrieve information.



The **Storage management** system or "database engine" ist the beating heart of the system. It is responsible to effect the actual transactions of the data: adding new data, modifying existing records, deleting data, or retrieving data according to complex relationships. In a multi user system it is a major challenege to implement such transactions in such a way that guarentees users' requests don't interfere with each other, and guarantees the integrity of the database even if a transaction fais midway because the sytem crashes. This is achieved by implementing the four **ACID requirements**:

A for "Atomicity" requires that transactions either completely succeed, or completely fail. No partial transactions can be possible that might bring the database into an inconsistent state. Transactions are not divisible.

C for "Consistency" means that every transaction will bring the database from one valid state to another valid state. All rules that have been defined must be respected at all times.

I for "Isolation" means that if transactions are performed concurrently in a multi-user system, the final state of the database must be the same as if they had been executed sequentially. Transactions can't overwrite each other during execution.

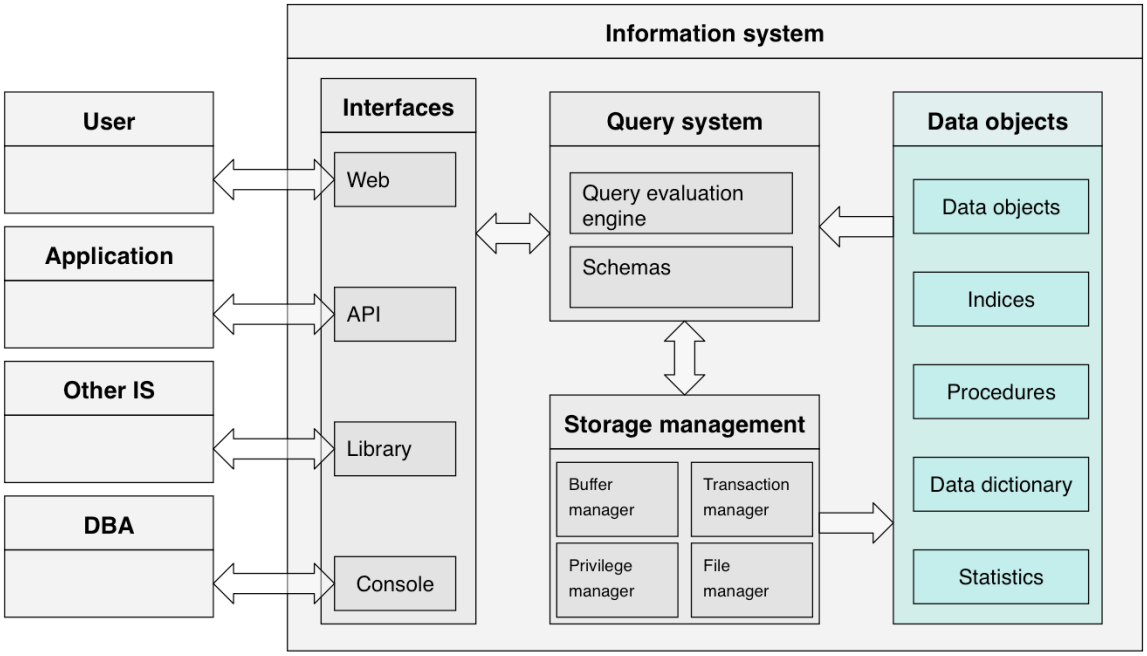
D for "Durability" means that the system tolerates crashes, power losses etc. – once a transaction has been succcessfully commited, it will remain committed.

Don't use database systems that are not ACID compliant. MySQL and PostgreSQL are ACID compliant.

Our learning units database is by its nature a single-user system, and all additions, deletions and modifications are done manually by editing the underlying JSON files. Retrieval is done by subsetting,

DATABASES

"Database" can refer to the actual data stored, but more generally a "Database" is a layered system of components to store, update and retrieve information.



Last not least, the actual data. This may include

the core **data objects**;

index tables that allow very fast, scalable retrieval of data items;

"**stored procedures**" – i.e. scripts that perform complex database operations;

a **data dictionary** that describes the nature of the tables, holds user login and privileges information, and records key configuration setttings;

and **statistics** about the table contents and use.

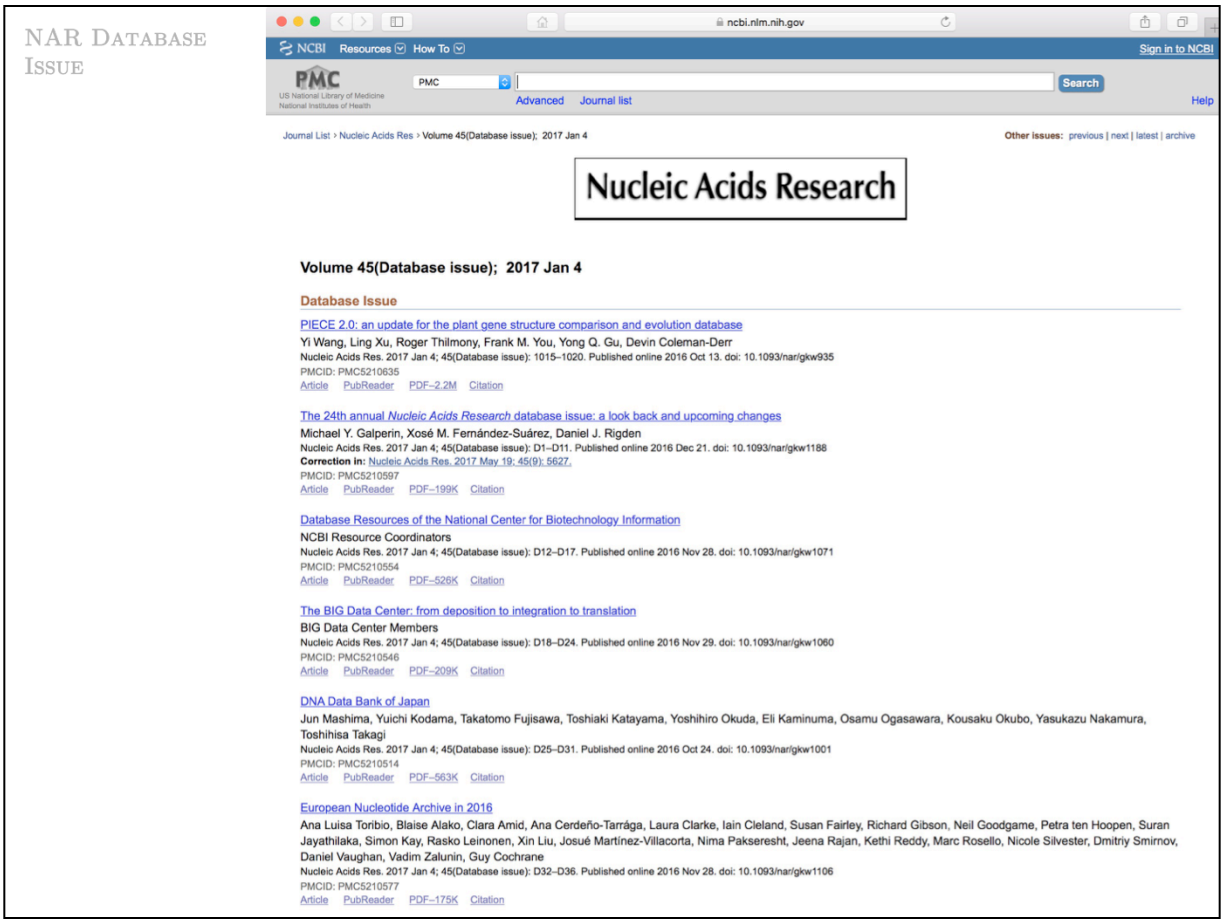
In our R learning units, the data is held in JSON files, made active as a list of data frames and thus consists of the raw data objects only.

DATABASES

- **Primary** Databases are **archives** of experimental results
  - GenBank – nucleic acid sequences
  - dbEST – expressed sequence tags (mRNA fragments)
  - PDB – macromolecular structure
  - PubMed – publications
- **Secondary** Databases **curate** compilations and interpretations
  - SwissProt – protein sequence + annotation
  - RefSeq – a non-redundant, annotated set of sequences at Genbank
  - UniGene – clusters of ESTs
  - PDBsum – structure annotations
  - KEGG – pathways curated from review articles
  - OMIM – genetic disease reviews

We used to discuss the difference between archives and curated compilations – but the boundaries have become fluid, and in terms of user experience there is really no difference. This slide is being kept mainly for historic reasons.

NAR Database Issue



NAR publishes a special issue on biological databases every year. Have a look. 158 articles in the 2017 edition.

http://steipe.biochemistry.utoronto.ca/abc

B O R I S . S T E I P E @ U T O R O N T O . C A

DEPARTMENT OF BIOCHEMISTRY & DEPARTMENT OF MOLECULAR GENETICS
UNIVERSITY OF TORONTO, CANADA