Data Management and Warehousing

Introduction

Stéphane Bressan

Contact

Stéphane Bressan

COM1-03-20

6516 3543

steph@nus.edu.sg

We want to develop a sales analysis application for our online gaming store. We would like to store several items of information about our customers: their first name, last name, date of birth, e-mail, date and country of registration on our online sales service and the customer identifier that they have chosen . We also want to manage the list of our products, the games, their name, their version and their price. The price is fixed for each version of each game. Finally, our customers buy and download games. So we must remember which version of which game each client has downloaded. It is not important to keep the download date for this application.

1. Go to `www.sqlite.org`
2. Go to `www.sqlite.org/download.html`
3. Download the command-line shell for accessing and modifying SQLite databases ("`A bundle of …`")
4. Extract the executable
5. You will find a short documentation at www.sqlite.org/sqlite.html

```
› .open bt5110.db
› .mode column
› .headers on
› .help
› ...
› ...

› .quit
```

You may skip the "`.open bt5110.db`" for now. That step creates a persistent database but also may slow down some update operations.

Let us create a table to store customer information. Let us call this table "`customers`". Each record in the table represents a customer. Each record contains a field to record the customer’s first name, "`first_name`", a field for saving the last name of the customer, "`last_name`", a field for saving the customer’s email, "`email`", a field for saving the customer’s date of birth, "`dob`", a field to record the customer's registration date, "`since`", a field to record the customer’s identifier, "`customerid`" and finally a field to register the country of registration of the customer, "`country`".

The SQL "CREATE TABLE" command creates the table. The name of the table and the name and domain (type) of each field must be specified: string of variable length, "VARCHAR ( )" and date, "DATE".

```
CREATE TABLE customers (
     first_name VARCHAR(64),
     last_name VARCHAR(64) ,
     email VARCHAR(64),
     dob DATE,
     since DATE,
     customerid VARCHAR(16) ,
     country VARCHAR(16));
```

Let us create a table to save the information about the games. Let us call this table "`games`". Each record in this table represents a version of a game. It contains a field to save the name of the game, "`name`", a field to save the version of the game, "`version`" and a field to record the sale price of the game, "`price`".

We see two new domains: fixed character strings "`CHAR()`" and "`NUMERIC`" numbers. There are many others.

```
CREATE TABLE games (
     name VARCHAR(32),
     version CHAR(3),
     price NUMERIC );
```

Let us create a table to save information about downloads. Let us call this table "`downloads`". Each record in this table represents the download of a version of a game by a customer. It contains a field to record the customer ID, "`customerid`", a field to save the name of the game, "name" and a field to save the version of the game, "`version`".

Note that these three fields have the same names and domains as the corresponding fields in the "`games`" and "`customers`" tables.

```
CREATE TABLE downloads (
     customerid VARCHAR(16),
     name VARCHAR(32),
     version CHAR(3));
```

We can now start inserting data. Let us create a record for the customer Carole Yoga. Carole was born on August 1, 1989. She registered with our online service on September 15, 2016 from France with the identifier "`Carole89`".

```
INSERT INTO customers VALUES(
     'Carole',
     'Yoga',
     'cyoga@glarge.org',
     '1989-08-01',
     '2016-09-15',
     'Carole89',
     'France');
```

The files below contain the SQL commands for creating the "`customers`", "`games`" and "`downloads`" tables as well as the data insertion commands, respectively.

`customers.sql`

`games.sql`

`downloads.sql`

```
CREATE TABLE downloads(
        customerid VARCHAR(16),
        name VARCHAR(32),
        version CHAR(3));

INSERT INTO downloads VALUES ('Adam1983', 'Biodex', '1.0');
INSERT INTO downloads VALUES ('Adam1983', 'Domainer', '2.1');
```

Excerpt from the file `downloads.sql`

```
› DROP TABLE downloads;
› DROP TABLE customers;
› DROP TABLE games;
› .read customers.sql
› .read games.sql
› .read downloads.sql
› INSERT INTO customers VALUES(
› 'Carole', 'Yoga', 'cyoga@glarge.org', '1989-08-01',
› '2016-09-15', 'Carole89', 'France');
›
```

A simple SQL query must include the "`SELECT`" clause, which specifies the fields to be printed, the "`FROM`" clause, which indicates the tables to be queried, and possibly the "`WHERE`" clause, which contains a possible condition on the records to be considered.
For example, the following query displays the first and last names of registered customers from Singapore.

```
SELECT first_name, last_name
FROM customers
WHERE country = 'Singapore';
```

Why are we here?

Data Base Management Systems (DBMS) are platforms that facilitate the design, development, deployment and maintenance of database applications. Relational database management systems (RDBMS) are platforms that provide a database (relational) data model.

The basic data management systems are to the files what a car's automatic driving is to manual driving.

Why file systems, or even tools like Excel, can cause problems to manage data?

Persistence must be managed explicitly.

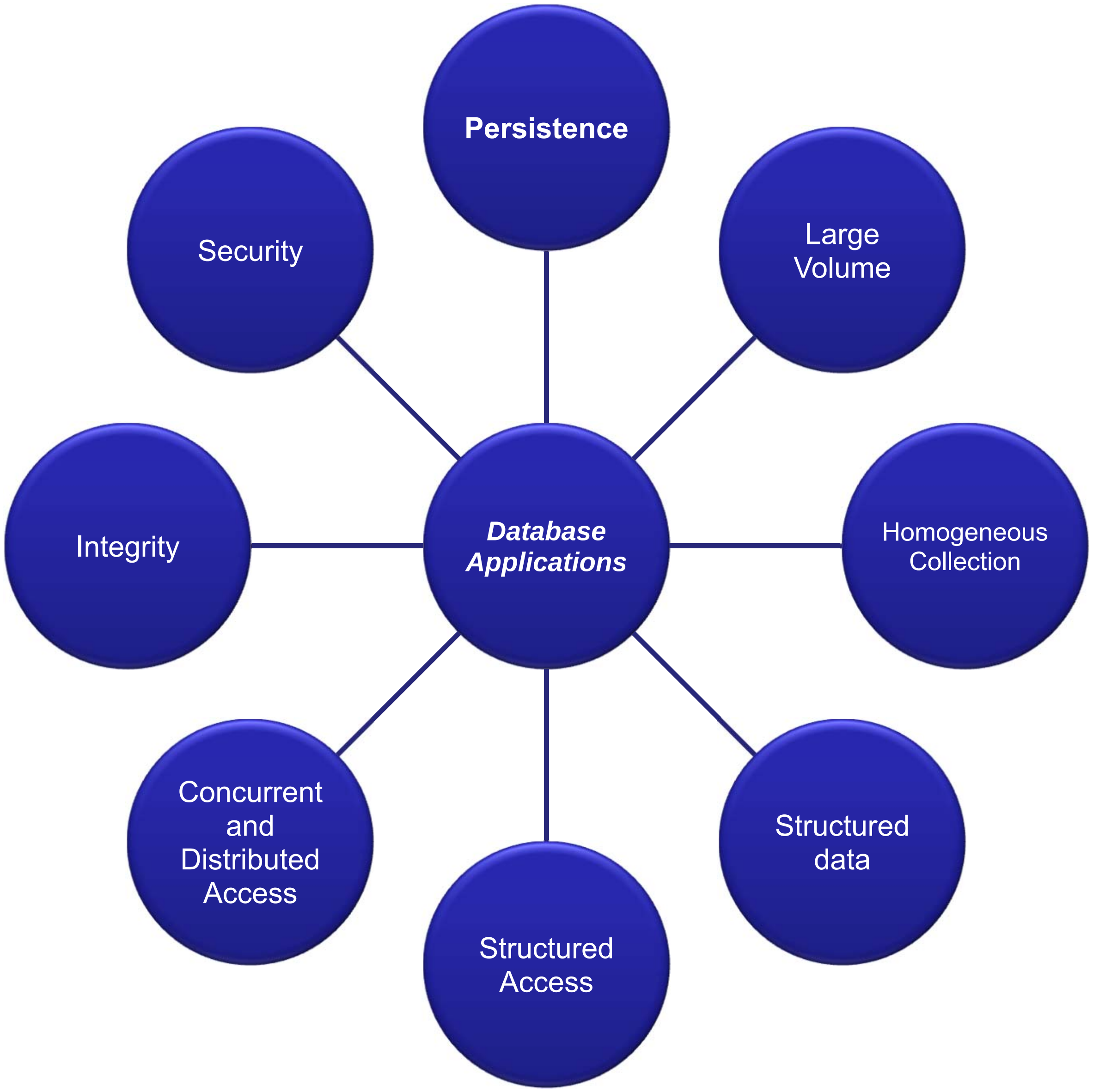
Updating and querying are difficult. Each processing requires a series of operations dependent on the organization of the data in the files which is neither necessarily homogeneous nor necessarily structured.

The organization of data for presentation is often redundant.

Concurrent and distributed access, integrity and security are difficult to manage.

Input errors, hardware errors, system errors or application errors leave the data corrupt and inconsistent.

Relational database management systems (RDBMS) provide solutions to solve or avoid these problems



Typical properties and requirements of database applications

DATA MUST PERSIST

How can data survive the process that created it, and be reused by other processes?

Mesopotamian tablet with cuneiform script to record crop stocks, taxes and laws from around 3200 BC

Large Amounts of Data

There are 180 million registered voters for the 2014 Indonesian elections

Where could one store the names, identificatio
electoral districts of voters?

Using www.matisse.net/bitcalc, evaluate the storage needed to store information about the 2014 Indonesian elections.

Large Amounts of Data

There are 180 million voters in the 2014 Indonesian elections

How could one sort them by alphabetical order of electoral districts and names?

Homogenous Collections of Structured Data

Relational Model

id	name	price	postal	street	Blk/no
257981	DELMONTE TOMATO KETCHUP	1.10	560215	Ang Mo Kio Ave 1	Blk 215
258021	DIAMOND ALUMINUM FOIL - 8.33YDS X 12IN	2.00	579837	Bishan Place	9
257835	/TEAPOT SWEETENED CREAMER #	1.05	7720	2 Ang Mo Kio Drive	Blk A

SQL

DDL: Data Definition Language. It includes statements to define the schema

DML: Data Manipulation Language. It includes statements for creating, updating, and querying data

DCL: Database Control Language. It include statements to administer access privileges and transactions properties

Transactions

A transaction is a logical unit of work carried out by a user or an application

Integrity of Data

How to maintain the integrity of data in spite of possible application, system, or media failures?

Consistent States

A consistent state of the database is a state which complies with the business rules as defined by integrity constraints

ACID Properties

Recovery

- **Atomicity: all actions in a transaction happen or none happen**
- **Durability: effects of successful transactions last**

Concurrency Control

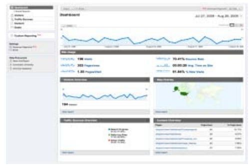
- **Isolation: Transactions can be understood independently from each other**
- **Consistency: If individual transactions would leave the application in a consistent state, a concurrent execution should do the same**

Access should be Controlled

Can you give examples of applications typically requiring a database?

The management of the accounts of a retail bank,
Ticket bookings of one or more airlines,
The management of a university,
The management of an online store,
Managing my address book ...

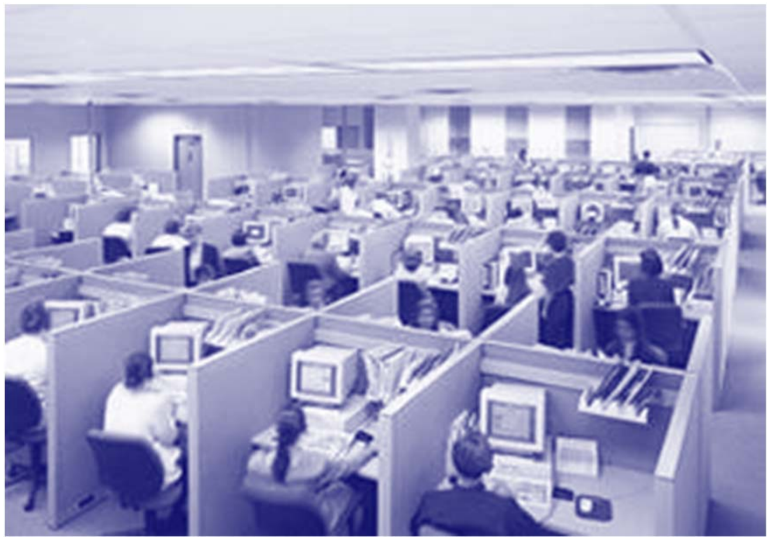
What operations do these applications perform on the database?
What properties these applications require from the database?



There are two main categories of database applications: transactional applications (we talk about online transaction processing or OLTP) and analytical applications (we talk about online analytical processing or OLAP) . The term "database" is typically used for transactional applications and the expression "data warehouse" is reserved for analytical applications. The difference is mainly due to the respective specifications and limitations of the technology. It is difficult to organize data access so that both complex updates and queries are fast. Great efforts are made by researchers and manufacturers to invent and develop technologies that allow the convergence of both types of applications.

Users

OLTP



OLAP

Utilization

OLAP

The utilization of an OLTP system is defined by the business process it supports.

OLTP

The utilization of an OLAP system is exploratory and evolving.

Workload

OLTP

Online transaction processing applications synchronizes and document business operations.

- frequent updates
- numerous transactions
- short transactions
- involve tiny portion of the data
- simple queries

Most Database Management Systems were designed for OLTP applications.

OLAP

Online analytical processing applications support the exploration and analysis of historical data for the purpose of model design and decision making.

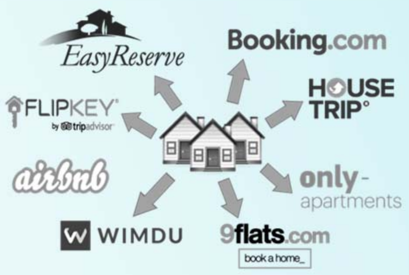
- mostly append only insertions
- scheduled, infrequent updates
- long transaction
- complex queries
- involve huge portions (all) the data

Modern Database Management Systems are tuned for OLAP applications.

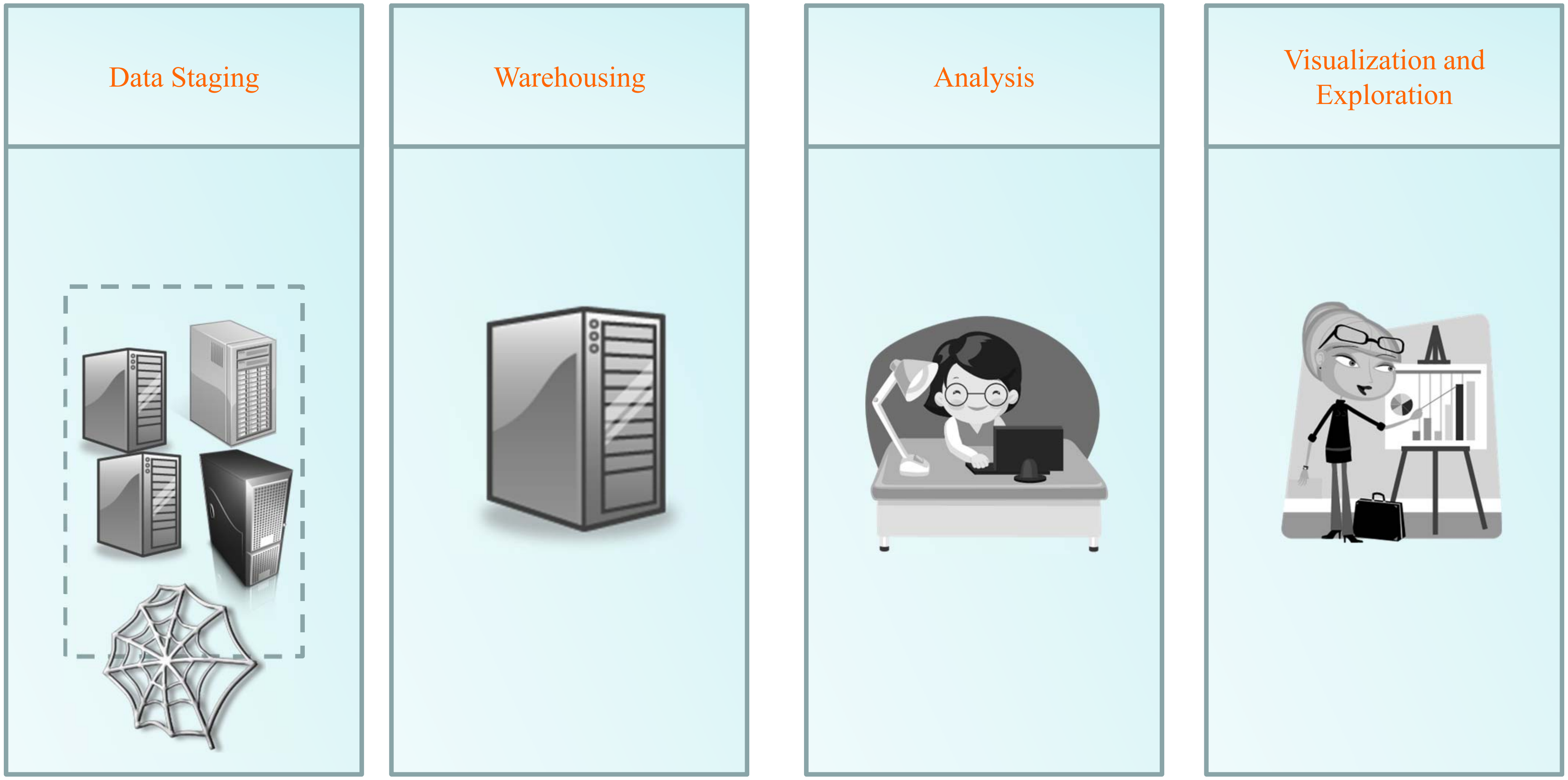
OLTP Architecture

Database

Application Server



OLAP Architecture



We study:

Relational Database Design

- Design with the entity-relationship model and normalization
- Design with SQL and creation of tables with integrity constraints

Creation, Update, Deletion and Querying of Relational Databases

- Simple and Complex SQL Statements and Queries
- Stored Procedures and Triggers

Data Warehouse Application Design

- Dimension Modelling and Star Schemas
- Dimension Modelling Case Studies

Creation and Querying of Data Warehouses

- ETL
- OLAP and Cubes Queries
- Reporting and Analysis with dashboards

Some bibliographical and Web references

"SQL Tutorial" www.w3schools.com/SQL

"Introduction to Databases" by Jennifer Widom, www.youtube.com or online.stanford.edu

"Database Management Systems " by R. Ramakrishnan et J. Gehrke , McGraw Hill

"An Introduction to Database Systems (8th edition)" by J.C. Date, Pearson

"The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling" by Ralph Kimball and Margy Ross, Wiley

Load and Assessment

- **Load:**

Lectures:	2 hours per week (13 Weeks)
Tutorials :	1 hours per week (10 Weeks)
Homework and Projects:	6 hours per week (at home)
Preparatory work:	1 hours per week (at home)

- **Assessment:**

Test 1 (Week 7):	25%
Test 2 (Week 13):	25%
Homework and Projects:	50%

Credits

steph@nus.edu.sg