Introduction to Databases

Prof. Joseph G Vella ©
Dept. of Computer Information Systems
JOSEPH.G.VELLA@UM.EDU.MT

1

Introduction to Databases

Rationale & Challenges

2

Where can we start!?

- **Definition:**
  - A **database** is a collection of structured data stored on persistent storage.
- **Environment and Contextual**
  - **Available** to a large number of end users.
    - **As in "Access my data when I need it …"**
      - **No delay, no access denial, no loss of data**
  - **Correctness**.
    - **As in "What's my balance?" gets the same value until it is changed.**
- **What's different from, say, programming?**
  - Data **persistence**;
  - Data **sharing**;
  - Data **independence**.

3

Example Database Instance (From *Elmasri*)

- **A lot of databases are built with a particular data modelling language called relational.**
- **This is an example of a relational database:**
  - It is built by set of **relations** (e.g., tables).
  - A group of tables are sometimes grouped in a **schema**
- **There are two aspects immediately visible:**
  - **Structures**;
  - **Data** (fitting the given structure).

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

4

Database schema

Structures built by a relational data modelling

STUDENT	Name	StudentNumber	Class	Major
	Smith	17	1	CS
	Brown	8	2	CS

COURSE	CourseName	CourseNumber	CreditHours	Department
	Intro to Computer Science	CS1310	4	CS
	Data Structures	CS3320	4	CS
	Discrete Mathematics	MATH2410	3	MATH
	Database	CS3380	3	CS

SECTION	SectionIdentifier	CourseNumber	Semester	Year	Instructor
	85	MATH2410	Fall	98	King
	92	CS1310	Fall	98	Anderson
	102	CS3320	Spring	99	Knuth
	112	MATH2410	Fall	99	Chang
	119	CS1310	Fall	99	Anderson
	135	CS3380	Fall	99	Stone

GRADE_REPORT	StudentNumber	SectionIdentifier	Grade
	17	112	B
	17	119	C
	8	85	A
	8	92	A
	8	102	B
	8	135	A

PREREQUISITE	CourseNumber	PrerequisiteNumber
	CS3380	CS3320
	CS3380	MATH2410
	CS3320	CS1310

THE END!

Joseph Vella - Introduction to databases 5 Introduction

5

Database instance

Data fitting respective relations

STUDENT	Name	StudentNumber	Class	Major
	Smith	17	1	CS
	Brown	8	2	CS

COURSE	CourseName	CourseNumber	CreditHours	Department
	Intro to Computer Science	CS1310	4	CS
	Data Structures	CS3320	4	CS
	Discrete Mathematics	MATH2410	3	MATH
	Database	CS3380	3	CS

SECTION	SectionIdentifier	CourseNumber	Semester	Year	Instructor
	85	MATH2410	Fall	98	King
	92	CS1310	Fall	98	Anderson
	102	CS3320	Spring	99	Knuth
	112	MATH2410	Fall	99	Chang
	119	CS1310	Fall	99	Anderson
	135	CS3380	Fall	99	Stone

GRADE_REPORT	StudentNumber	SectionIdentifier	Grade
	17	112	B
	17	119	C
	8	85	A
	8	92	A
	8	102	B
	8	135	A

PREREQUISITE	CourseNumber	PrerequisiteNumber
	CS3380	CS3320
	CS3380	MATH2410
	CS3320	CS1310

THE END!

Joseph Vella - Introduction to databases 6 Introduction

6

Example Database Structure and Data (From *Elmasri*)

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

- **Note:**
  - Table structure composition (e.g., attribute name and data type (assumed));
  - Table's data;
  - Are there any other rules which the data must follow?
    - **Primary key, limiting possible values in a data type, foreign key, etc.**

7

For what do we use a database? How do we use a database?

- **Typical *operations* over a database are:**
  - *accepting* structured data (*ADD*, *AMEND* and *PURGE/DELETE*).

- **How do we use databases?**
  - *computing* functions on the present data;
  - *sorting* the structured data; and
  - *providing* responses to questions (querying).

8

How is usage managed and controlled?

- **A Database Management System (DBMS)**
  - Is a computer-based application tool set for: defining, creating, manipulating, controlling, and managing databases.
- **DBMS Requirements:**
  End users, developers, administrators demand efficiency in terms of allocation of computational resources and storage space.
  - **Reliability** (e.g., if the system falls, can it relive itself)
  - **Openness** (e.g., in term of data connectivity)
  - **Scalability** (e.g., storage and throughput)

9

DBMS Functionality

- **DBMS also facilitate and insulates data access by system's end users (including "computer" naive *end users*).**
- **Three important functionalities include:**
  - *Query Processing*
  - *Transaction Management*
    - *In some context TP subsumes QP*
  - *Storage management*
- **Two main areas of concern:**
  - *Consistency* (e.g., returns the same results under invariant states), and
  - *Efficiency* (e.g., computational time and space) of DBMS activities.

10

DBMS Advantages

- **Some advantages:**
  - Execute a query in an efficient and opportunistic approach – **query optimisation;**
  - Iron out sharing inconsistencies between conflicting queries and transactions – **concurrency control;**
  - Undoing the effects of incomplete transactions under a system's instructions or in rectifying a system failure – **recovery.**
    - **Note: system here is most general: OS, DBMS, hard disk, etc.**

11

Query Processing (and Optimisation)

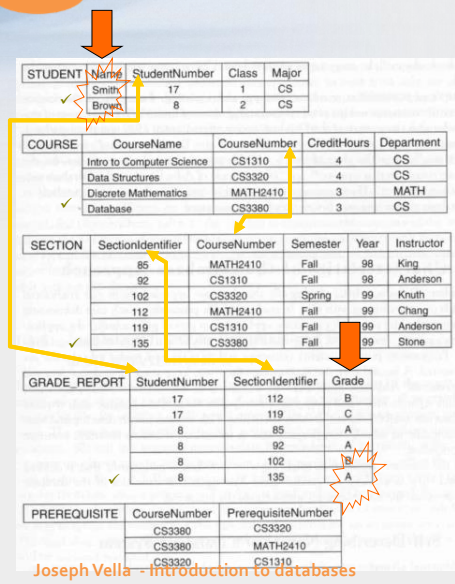
12

Query Processing & Optimisation

- We now briefly describe the structure and functionality of a *query processor* -- a process that transforms and executes a user-defined query against a database state and returns those instances that satisfy the query.
- A *declarative query* is a data access expression that indicates the structure and the property of the result (e.g. what you want). In contrast to a procedural query that specifies structure, property and an exact computational program(e.g. how to get).
  - Efficiency of a ***declarative relational query processor*** (for languages such as **SQL**) is a major project.
    - *Query optimisation* is the tuning of queries and storage structures to favour the performance of *some* frequently used query.
- What is a data access path?
  - A procedure to traverse and access required data items through the use of available look-up methods associated to the data file organisation.

13

QP Example:
Return all student's name and grade in the database unit.



Step One: Identify the attributes to output
i.e. Student / Name & Grade_report / Grade.

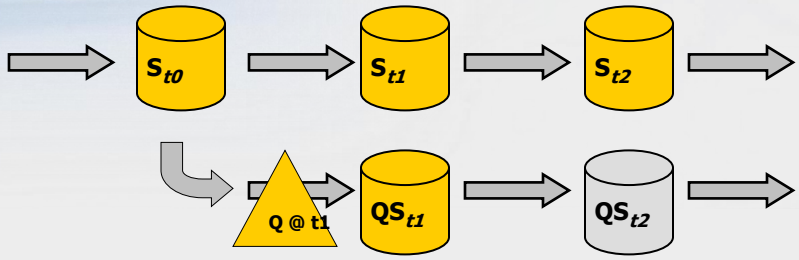
Step Two: Identify the relationships between structures of interest to the query
i.e. a) Student / StudentNumber to Grade_report / StudentNumber,
b) Course / CourseNumber to Section / CourseNumber,
c) Section / SectionIdentifier to Grade_report / SectionIdentifier.

Step Three: Apply any restrictions
i.e. Course / Coursename is equal to 'Database'.

Step Four: Compute the output.
i.e. Student 'Brown' with a grade of 'A'.

14

QP while the database is available (to other users)



At time t1, query Q's state (i.e. result QS) is related to the database state at t0.

Database states at t0 and t1 are equivalent (i.e. St0 = St1).

Database states at t1 and t2 are not necessary equivalent (i.e. St2 <> St1).

At time t2, query Q's state (i.e. result QS at t2) is NOT related to the database sate at t2.

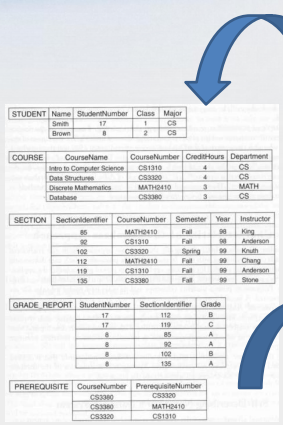
Query Q states at t1 and t2 are equivalent (i.e. QSt1 = QSt2).

The end!

Joseph Vella - Introduction to databases 15 Introduction

15

QP Example – another view



STUDENT	Name	StudentNumber	Class	Major
	Smith	17	1	CS
	Brown	8	2	CS

COURSE	CourseName	CourseNumber	CreditHours	Department
	Intro to Computer Science	CS1310	4	CS
	Data Structures	CS3320	4	CS
	Discrete Mathematics	MATH2410	3	MATH
	Database	CS3380	3	CS

SECTION	SectionIdentifier	CourseNumber	Semester	Year	Instructor
	85	MATH2410	Fall	98	King
	92	CS1310	Fall	98	Anderson
	102	CS3320	Spring	99	Knuth
	112	MATH2410	Fall	99	Chang
	119	CS1310	Fall	99	Anderson
	135	CS3380	Fall	99	Stone

GRADE_REPORT	StudentNumber	SectionIdentifier	Grade
	17	112	B
	17	119	C
	8	85	A
	8	92	A
	8	102	B
	8	135	A

PREREQUISITE	CourseNumber	PrerequisiteNumber
	CS3380	CS3320
	CS3380	MATH2410
	CS3320	CS1310

```
SELECT st.name, gr.grade
FROM   student st,
       course co,
       section sc,
       grade_report gr
WHERE  co.coursename='Database'
  AND st.studentnumber=gr.studentnumber
  AND gr.sectionidentifier=sc.sectionidentifier
  AND sc.coursenumber=co.coursenumber;
```

Q @ t1

S_{t0}

Name	Grade
Smith	B
Brown	A

QS_{t1}

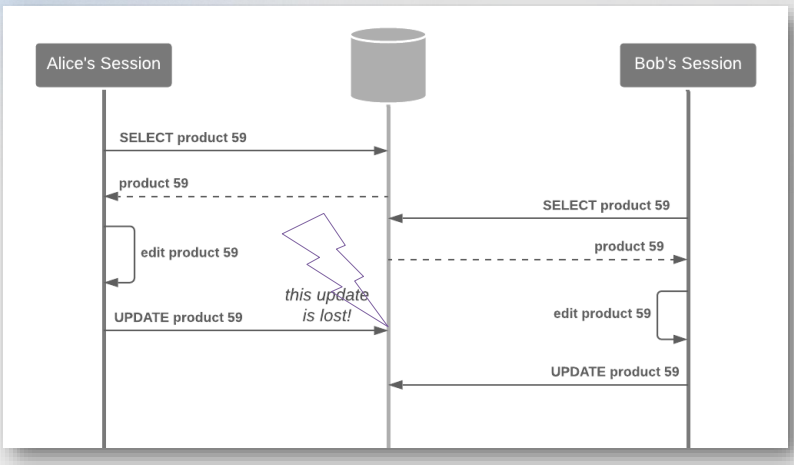
Joseph Vella - Introduction to databases 16 Introduction

16

Transaction Processing and Recovery

17

Sharing Issues – e.g., lost update type



18

Transaction Modelling

- **An unrestricted sharing over data is problematic:**
  - The example on the right is called ***dirty read*** as T2 reads what T1 has written. If Commit of T1 does not pass then T2 is wrong.
- **An explicit sequence of data manipulation commands over a database are abstracted as a single logical update (i.e a transaction).**
  - A transaction is **valid** if ***all*** its sub-parts (e.g. updates, deletes) have succeeded. A valid transaction can **commit,** otherwise we have to **rollback** the transaction.
- Operationally this complicates our scenario when allowing for data sharing! The DBMS typically insulates this from users (programmers and end users alike!).

$$S = \begin{bmatrix} T1 & T2 \\ R(A) & \\ W(A) & \\ & R(A) \\ & W(A) \\ & R(B) \\ & W(B) \\ & Com. \\ R(B) & \\ W(B) & \\ Com. & \end{bmatrix}$$

19

TP Principles - ACID

- **Transaction's ACID principles for short-lived transactions are** (over and above valid transactions):
  - **Atomicity**
    - **The transaction executes completely or not at all,**
  - **Consistency**
    - **The transaction preserves the internal consistency of the database.**
  - **Isolation**
    - **The transaction executes as if it were running alone, with no other transactions.**
  - **Durability**
    - **The transaction's result will not be lost in the future.**

20

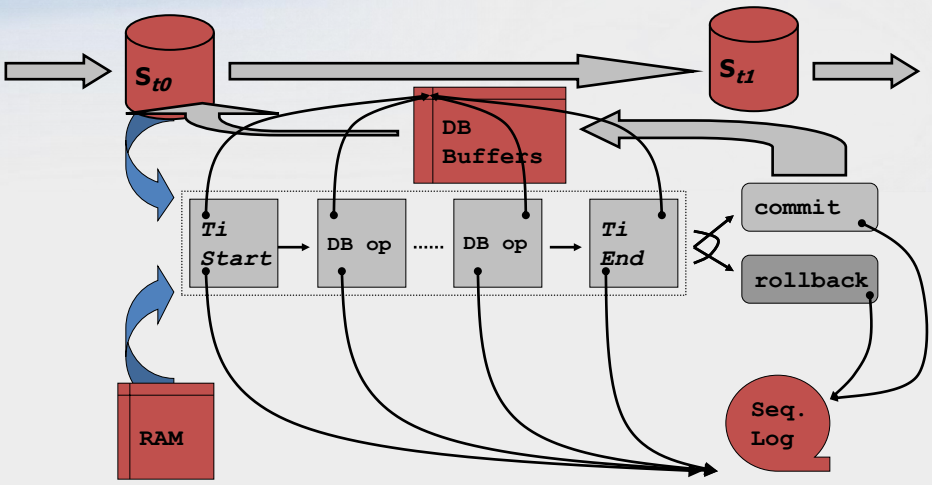
Transaction Modeling: an Example

- "Ensure that the employees' salary is upped by 25% and each salesperson commission is augmented by Euro 250."

```
Remark SQL command sequence
     start transaction;
       update  emp
          set  sal = sal * 1.25;
     remark implicit mechanism - on error rollback
       update  emp
          set  comm = comm + 250
        where  comm is not null;
     remark implicit mechanism - on error rollback
     commit;
     end transaction;
```


21

Transaction Modelling and Database States

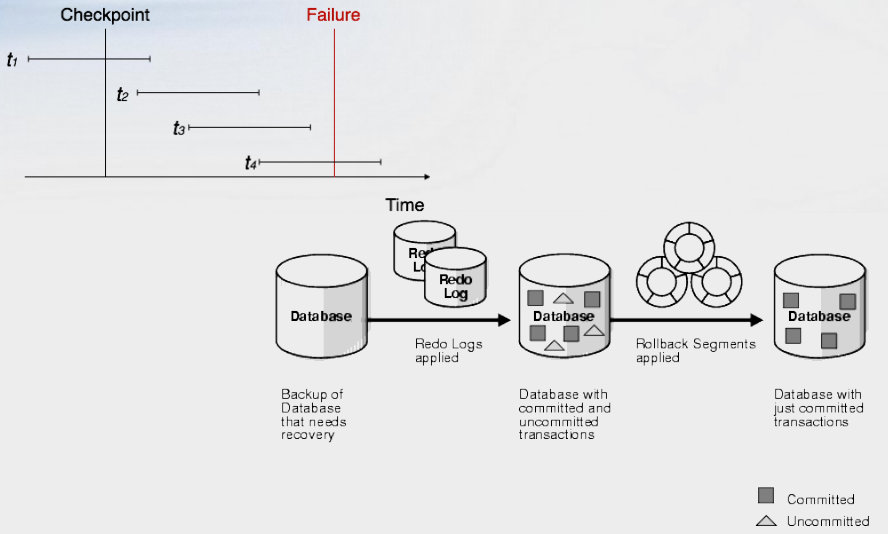


The end!

22

23

We need to restore database state to end of *t3*



Joseph Vella - Introduction to databases 24 Introduction

24

The Parts

25

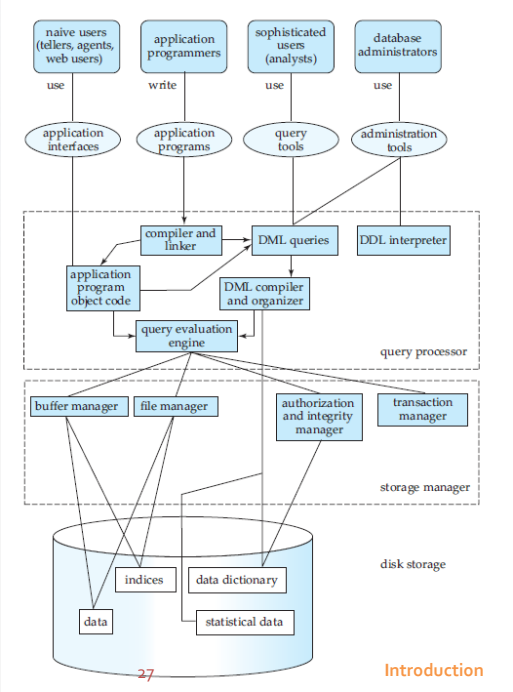
DBMS components

- Procedural and Declarative database programming languages;
- Database browsers;
- Administration and Data Dictionary tools;
- Database connectivity protocols (e.g. ODBC);
- Performance tuning tools;
- Import / Export Data utilities;
- CASE (and BPR) tools for conversion of a conceptual database design into a set of data modelling constructs;
- Forms and Report generators;

Joseph Vella - Introduction to databases 26 Introduction

26

Generic DBMS Architecture



Joseph Vella - Introduction to databases 27 Introduction

27

Available DBMS (a biased list 😐)

ORACLE®

SQLite

Microsoft® SQL Server® 2008

VERTICA

PostgreSQL

IBM DB2

neo4j

mongoDB

Joseph Vella - Introduction to databases 29 Introduction

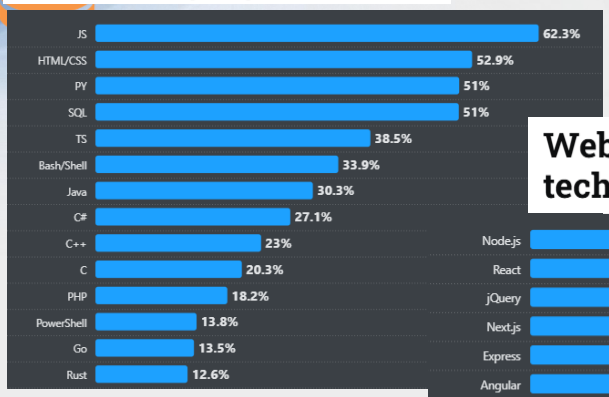
29

Available DBMS (a biased collective list) Stackoverflow 2024 ()

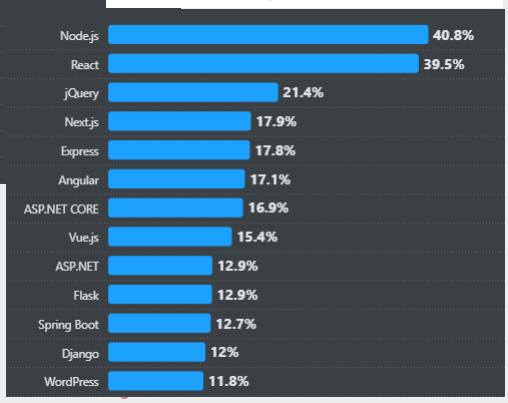
30

Programming, scripting, and markup languages

2024

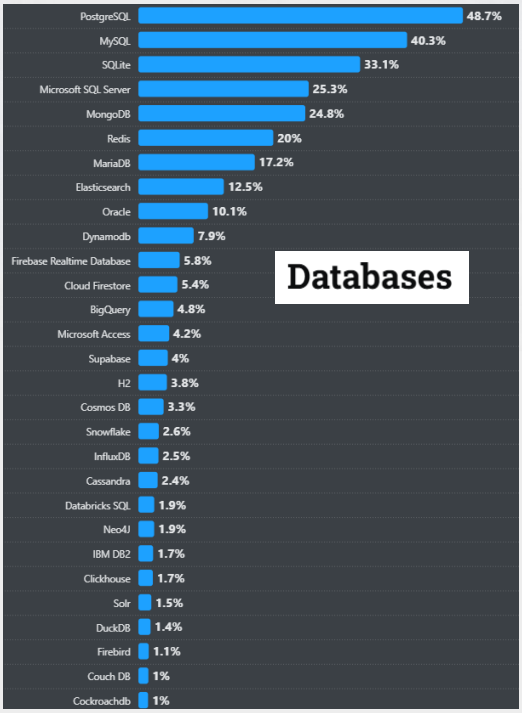


Web frameworks and technologies



31

2024

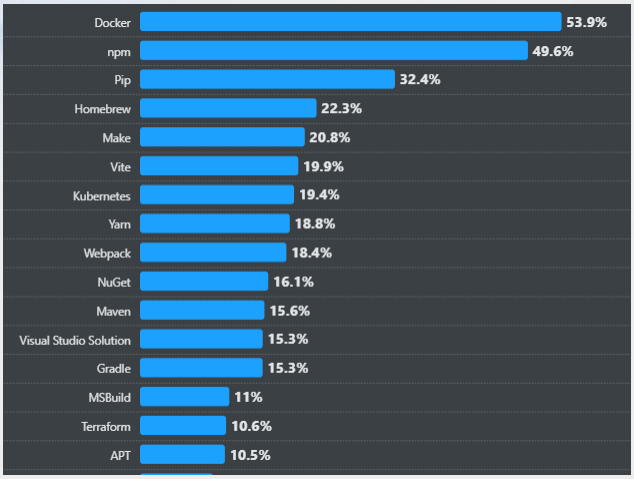


Joseph Vella - Introduction to databases

32

Other tools

2024



Joseph Vella - Introduction to databases 33 Introduction

33

Old copies

Just to keep the older copies

34

Available DBMS (a biased collective list) Stackoverflow 2022 (June)

https://cloud7.news/development/stack-overflow-published-its-developer-survey-2022-results/

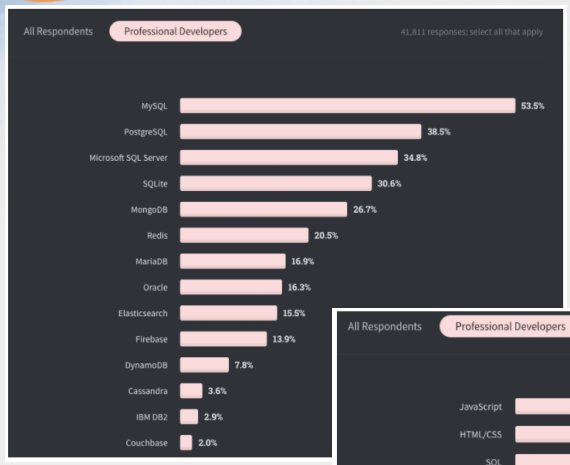
35

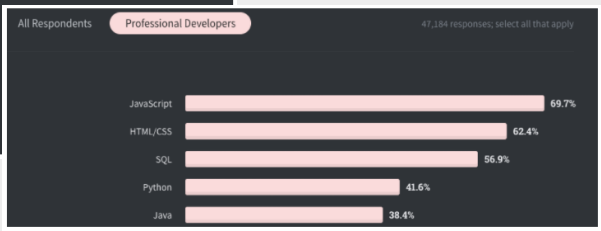
Available DBMS (a biased collective list)
Stackoverflow 2022 (June)

https://cloud7.news/development/stack-overflow-published-its-developer-survey-2022-results/

36

Available DBMS (a biased collective list 😐)
Stackoverflow 2020 (June)





37

The Key Ideas

38

Keys ideas (for this unit)

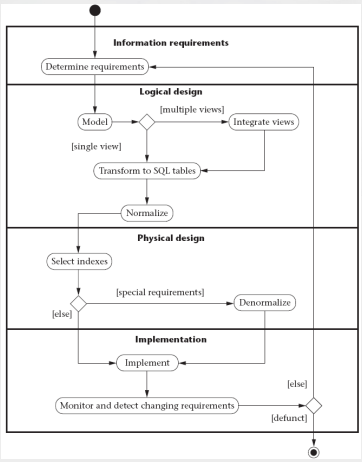
- Data Modelling
  - Build a database design to requirements (including data requirements).
- **Schema and data management**
- Structured Query Language (SQL):
  - Data definition language (DDL)
  - Data manipulation (DML)
    - ***Query modelling*** **as in read only operations;**
    - ***Transaction modelling*** **as in read & write operations.**
  - Rule definitions
    - ***Integrity constraints***
    - **(and *database triggers*)**
- **Elements of good design (e.g., data normalisation)**
- **Latching databases to programming exercises:**
  - Intra: stored procedures;
  - Inter: latching to programming languages.

Joseph Vella - Introduction to databases 39 Introduction

39

Database life cycle (i)

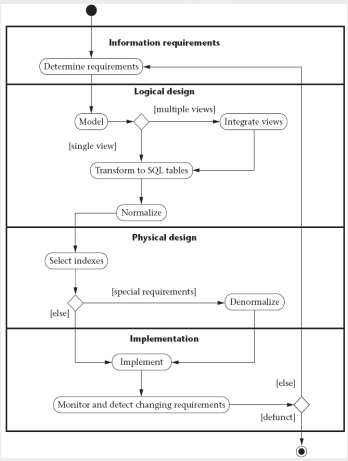
- **Get the requirements and extract database related information …**
  - *Do not shy from certain topics!* e.g., transaction processing, platforms, data security
- **It is best to start with the conceptual schema and continue with adding and integrating external views as required.**
  - These have to be supported from the requirements.
  - **Note:** external views are indispensable for application development (e.g. forms and reports) and data security basis.
- **When global and external views are available one translates them into a DBMS specific data model constructs.**



40

Database life cycle (ii)

- **Once DBMS specific artifacts are introduced one can start to introduce physical objects to address *performance*, *reliability* and *connectivity* issues.**
  - Again it's best to reconcile with requirements.
  - Physical artifacts include:
    - **Variety of indexes;**
    - **Data placement policies;**
    - **DBMS set-up parameters.**
- **Once the database is active, DBMS tools are used for monitoring performance and comparing this with expected guidelines.**



41

Database systems:
areas of interest for IT people (i)

- **Design of database schemas**
  - How to build a useful database?
  - What descriptions and relationships need to be depicted in a schema?
  - How can we integrate with other established databases?
  - How to capture and quality check external data?

42

Database systems:
areas of interest for IT people (ii)

- **Database programming**
  - How to access and control database 'connections'?
  - How to express pattern matching for data retrieval (query specifications)?
  - How to express state change instructions over the database (transaction specification) and control their execution?
  - How best to live with DBMS facilities (data persistence, integrity constraints, transaction processing)?
  - How to align database programming with front-end programming (e.g. GUI at client presentation)?

43

Database systems:
areas of interest for IT people (iii)

- **DBMS implementation**
  - What's available (e.g. indexing methods)?
  - What's configurable?
    - **e.g. can we overrule default B-tree with a hash index?**
  - Are there any indications on performance issues?
    - **e.g. in 95% of queries submitted their response time must be within 3 s.**
    - **e.g. 95% of submitted transaction have to commit.**
  - What are the tradeoffs between various components (e.g. storage, authorization, transaction and query processing)?

44

Guiding Principles

45

Database systems:
guiding principles (i)

ServiceCard

Serial Numb.	Service Type	Client Number	Client Name	Hours	Hrs to Charge
110	SERVICE	25010	Jeffrey	3	0
120	SERVICE	25150	Jeremey	2	2
130	REPAIR	25060	JJ	2	2
140	INSTALL	25090	Jaz	3	1
150	MA	25090	Jaz	2	0

- Consider the following data extracts.

ServiceCard

Serial Numb.	Service Type	Client Number	Hours	Hrs to Charge
110	SERVICE	25010	3	0
120	SERVICE	25150	2	2
130	REPAIR	25060	2	2
140	INSTALL	25090	3	1
150	MA	25090	2	0

ClientDetails

Client Numb.	Client Name
25010	Jeffrey
25150	Jeremey
25060	JJ
25090	Jaz

46

Database systems:
guiding principles (i)

- **Consider the previous slide with data extracts:**
  - Not in 3nf
  - And
  - Fixing it by decomposing into two tables.
- **Data redundancy (i.e. at Conceptual level)**
  - Is the need to ensure, to a high degree, that data in a database is not duplicated.
    - **If duplication (i.e. redundancy) exists then inconsistency arising from it has to be addressed (e.g. extra coding, obscures data meaning).**
- **We want to minimize data redundancy at logical level.**

47

Data Independence:
Motivation & Definitions

- **The *separation* of physical data items (e.g. files), logical data items (relational database), and application programs that interact with the database minimises their inter dependence.**
  - The concept of data independence conveys our wish to make artefact changes without unduly having to *re-map* (or recompile) all the higher artefacts that depend on the artefact.
    - **Lowest is the physical level , middle is the logical and higher is the AP.**
- **Logical data independence**
  - changes to the schema do not necessitate changes in the application programs. **For example: adding new data types - programmed access should still be consistent.**
- **Physical data independence**
  - changes to the physical schema are insulated from conceptual and application programs that use the database. **Typical physical changes include: index re-structuring; records re-ordering in a data file.**

Joseph Vella - Introduction to databases 48 Introduction

48

Data Independence:
ANSI-SPARC Architecture (3 level)

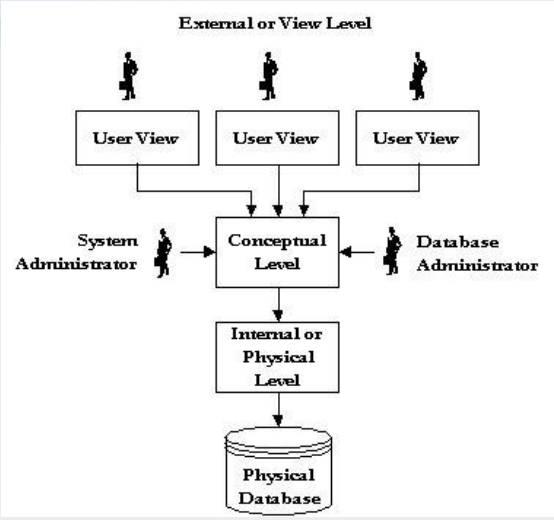
The classic representation of a database and DBMS data architecture is the ANSI/X3/SPARC proposal (1978). This architecture comprises three levels of abstraction.

- **Internal Schema**
  - This is the physical storage model containing information such as devices, file locations, structures, indexing systems and access methods.
- **Conceptual Schema**
  - A single and consistent model of the database of interest.
- **External Schemas**
  - A view of the conceptual model applicable users or processes.
    - **There are usually many of these and view overlapping is allowed.**

Joseph Vella - Introduction to databases 49 Introduction

49

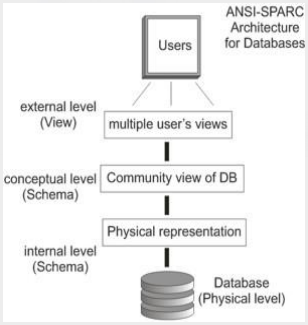
Data Independence: ANSI-SPARC Architecture (3 level)



50

Database systems: guiding principles (ii)

- **We want to minimize disruption due to schema changes.**
  - Change physical characteristics without the need to change the conceptual schema;
  - Change the external views without changing the conceptual schema;

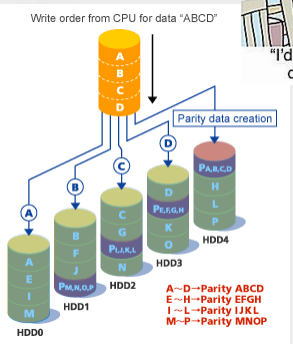


51

Database systems:
guiding principles (iii)

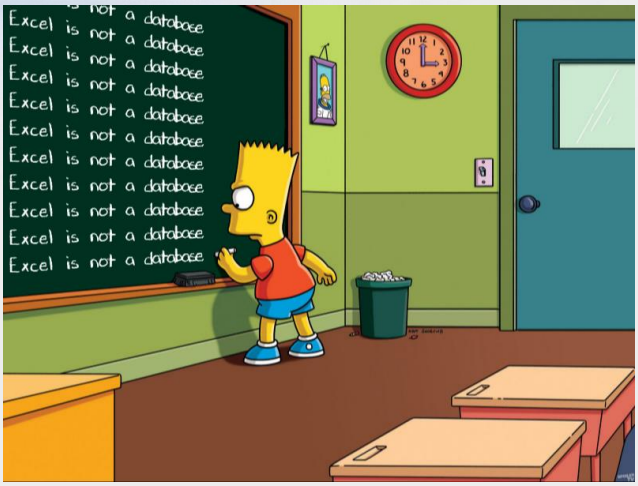
- **Data security**
  - We said earlier that Data is indispensable:
    - **Protect from 'abuse' – e.g. security breaches, errors, contention, h/w failures (e.g. hard disks fail).**

"I'd just like to know how these crazy rumors get started."

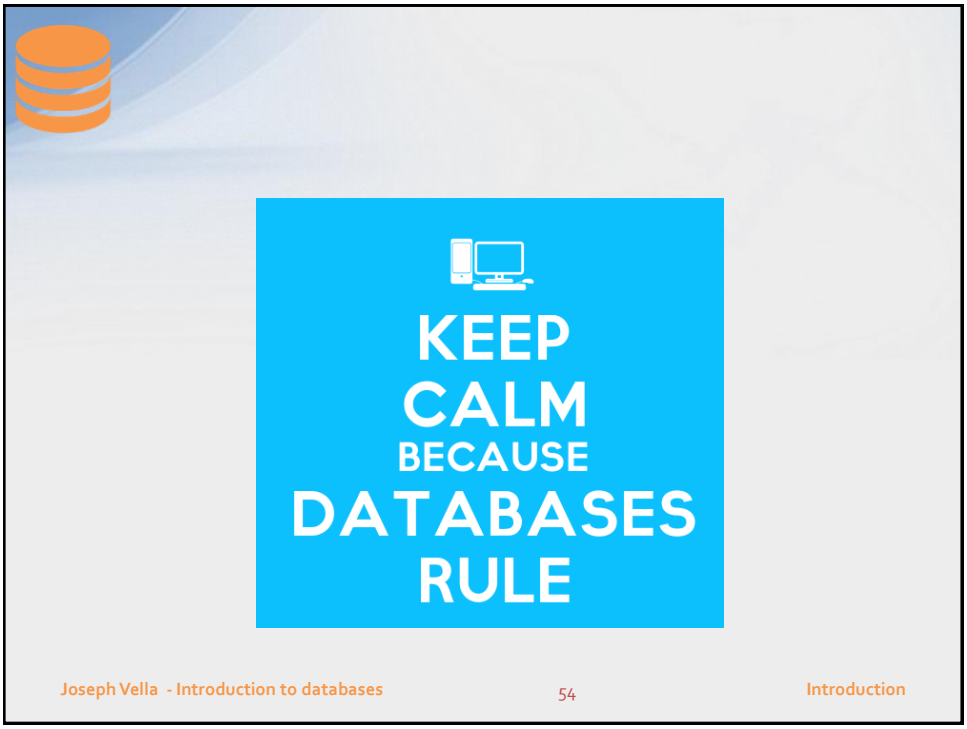


"I don't believe it!" – but true!? http://datacent.com/failing_hard_drive_sounds

52



53


KEEP
CALM
BECAUSE
DATABASES
RULE
Joseph Vella - Introduction to databases
54
Introduction

54