{Kariera Ruby Coding School - June 2017}

DATA MODELLING AND RELATIONAL DATABASES

A gentle introduction

Dr. John Pagonis

@zanshinlabs

Zanshin Labs

THE DONUTS ARE SPONSORED BY...

Zanshin
Labs

software production
consultants

we help teams & organisations

define | design | develop

software their **users need**

A STORY ABOUT DATA

Can you name one case where **data** are less important than the **algorithms** that use them?

Ancient computing proverb....

Garbage In Garbage Out

It doesn't matter how smart your algorithm is when you feed it with garbage....

Ancient computing proverb....

Garbage In Garbage Out

This also applies for your shiny new Rails app...

This is your app

Make sure that the kernel of your app is not rotten!!

What about Rails?

Rails is a Ruby framework for building **data(base) driven** web applications

It’s business as usual

Organisations need (IT) to perform their day-to-day **operations** and also make **decisions**

It has always been like this since the ancient times.

It’s business as usual

Organisations need (IT) to perform their day-to-day **operations** and also make **decisions**

To achieve this, they need **information**.

What is the difference between **data** and **information**?

Data vs information

Data is the collection of **raw facts**, whereas information is those data transformed in an **actionable** and **meaningful** form

---

i.e. Can you learn something? Can you do something?

How can we transform **data** into **information**?

To transform **data**
to **information**

Combine

To transform **data**
to **information**

Combine
Compare

To transform **data**
to **information**

Combine
Compare
Filter

To transform **data**
to **information**

Combine
Compare
Filter
Summarise

To transform **data** to **information**

Combine
Compare
Filter
Summarise
Organise

About transforming data to information

The more **flexible** you are in doing such transformations the **better**

The **more ways** you see data the **more info** you get!

About transforming data to information

The more **accurate** you are in doing such transformations the **better**

G.I.G.O

About transforming data to information

The **fastest** you are in doing such transformations the **better**

---

e.g., for event based decisions

It’s business as usual

Organisations need information to run their **operations** and make **decisions**

Provide and capture the right information to do the **job right**.

How **powerful**
(flexible, fast, accurate)
do you want to be at storing,
retrieving and producing
information?

ABOUT DATABASES

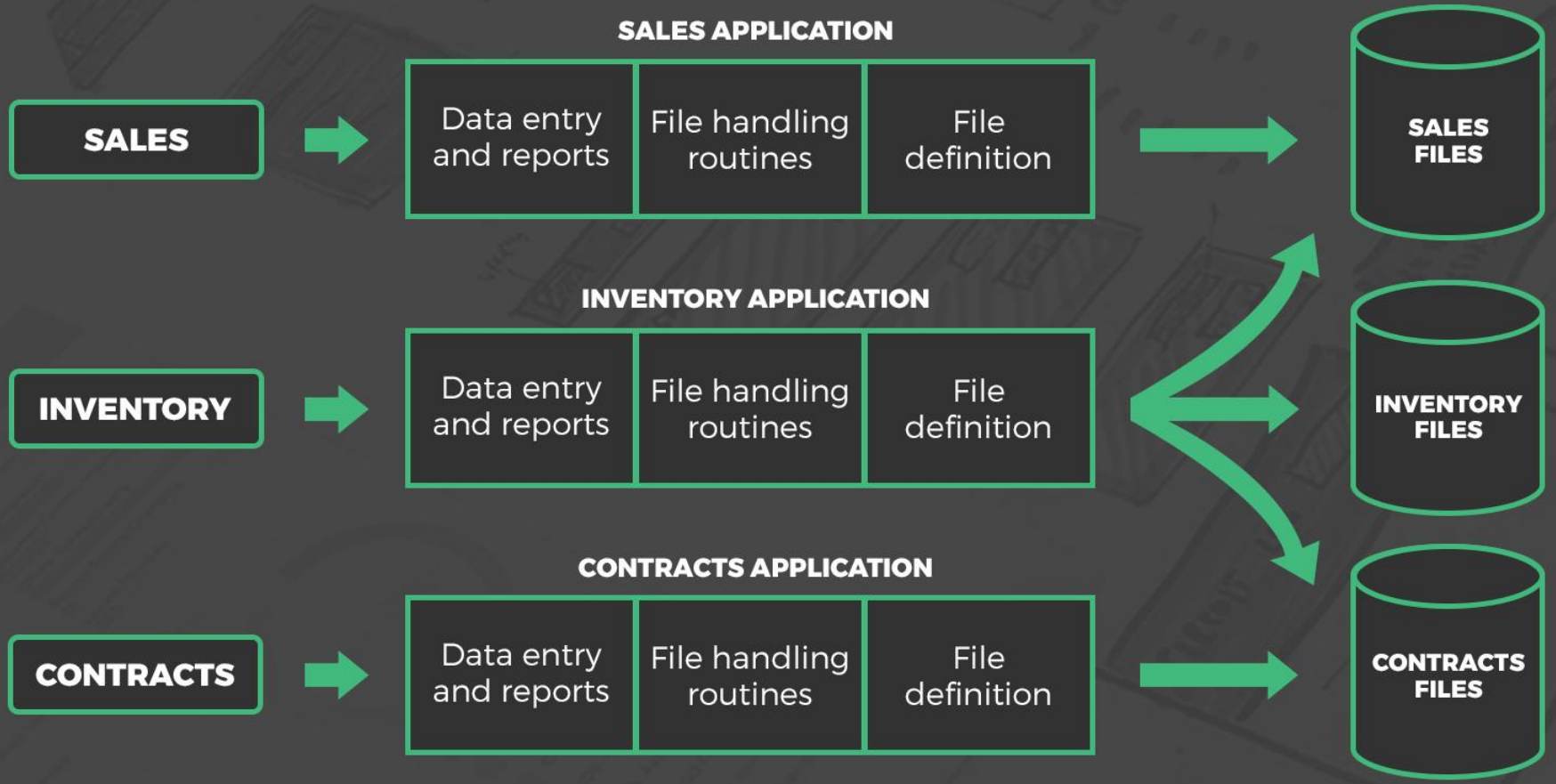
What used to happen (at organisations) before **databases**?

File-based processing...

We had **flat files** being accessed by many programs **directly**.

Each one of these had to know the exact **same rules** about the business and the data.

File-based processing...



Can you imagine any **problems** with that setup?

So what are **databases** then?

Databases

Collection of logically **related data** with descriptions about these data (i.e. **metadata**)

Metadata in databases provide
program-data independence (among other things).

Databases

Collection of logically **related data** with descriptions about these data (i.e. **metadata**)

In a database, conceptually (and many times physically) everything is stored **in one place**.

A DB is typically directly accessed by one program, the **DBMS**

The DBMS **enforces constraints** to the database.

So, is there a major difference between **MySQL** and **SQLite**?

A **client/server** DB is only directly accessed by one program, the **DBMS**.
It is the **gateway** through which other programs access the data

---

The DBMS **enforces constraints** to the database.

A taxonomy of databases (there are many taxonomies)

Hierarchical
Relational
Document-oriented
Object-oriented
Object-relational
Graph

RELATIONAL DATABASES

Relational databases (strictly speaking)

A relational database is one which is based on the relational model of data as defined by **E.F Codd**

Codd, E.F. (1970). "A Relational Model of Data for Large Shared Data Banks". Communications of the ACM. 13 (6): 377–387.

A relational database stores data about multiple different **entities**

Entities are stored in **tables**.

A relational database stores data about the **relationships** between these **entities**

The relationships are **links** between tables.

Relational databases (in simple terms)

A relational database is a collection of **entities** and the **relationships** between them

Relational databases (in simple terms)

A relational database is a collection of **entities** and the **relationships** between them

---

Each **table** should be about **one entity**.
Everything should only be there **once**.

Relational databases (in simple terms)

A table is a collection of data about one **entity**. It is a collection of **rows**. Each row has the same **columns**.

An **entity** is typically expressed and identified as a **noun**. If a table has data about more than one entity, then it has **redundant** data.

A column is an **attribute** of an entity. It is one elemental piece of data.

An **attribute** may also be expressed as a **noun**.

Relational databases (in simple terms)

A row is a collection of columns about **one member** of an entity.

An **attribute** may also be expressed as a **noun**.

A primary key is a special column (or set of) that **uniquely identifies** a row.

A primary key is a special column (or set of) that **uniquely identifies** a row. There can only be one per table. It can’t be null.

A primary key is a special column (or set of) that **uniquely identifies** a row. There can only be one per table. It can't be null.

If the entity has one then every row **has one**.

A primary key is a special column (or set of) that **uniquely identifies** a row. There can only be one per table. It can't be null.

If the entity has one then every row **has one**.
No two rows have the same one.

A primary key is a special column (or set of) that **uniquely identifies** a row. There can only be one per table. It can't be null.

If the entity has one then every row **has one**.
No two rows have the same one.
There are **natural** and **surrogate** primary keys.

A foreign key is an attribute in one table that **refers** to a primary key in another table.

So, how can we implement **relationships** in a **relational** database?

Relational databases (in simple terms)

We use **keys** to **link** tables together

A table's **primary key** is used to link other tables back to **itself**. It is then called the **foreign key** in the other tables.

LAB
1

Within your team

- Work in pairs on your project
- Identify the entities (look for nouns)
- Identify the relationships between the entities
- Identify the important attributes for each entity
- As a team outline on paper, if not all, at least the 5 most important entities and their relationships

Shall we **share?**

Within your team

- Is there something you can improve in your design?
- Why?

RELATIONAL DATABASES PART II

What happens if a key ends up pointing to a row which has been **deleted**?

Referential integrity (violation of)

Then, we say that we have **broken** the **referential integrity** of our database

Referential integrity (violation of)

Then, we say that we have **broken** the **referential integrity** of our database

---

For referential integrity to hold, a **foreign key** must reference a valid, existing primary key in another table.

Referential integrity (violation of)

Then, we say that we have **broken** the **referential integrity** of our database

---

For referential integrity to hold, a **foreign key** must reference a valid existing primary key in another table. Relational databases enforce **constraints** to make sure these keys match up.

How can we indicate in a relational database that a number of orders **belong to** a client?

Nature of relationships

- one to many
- many to one
- many to many
- one to one

Direct vs indirect relationships

Some of the relationships are just **derived** relationships and not **direct** relationships

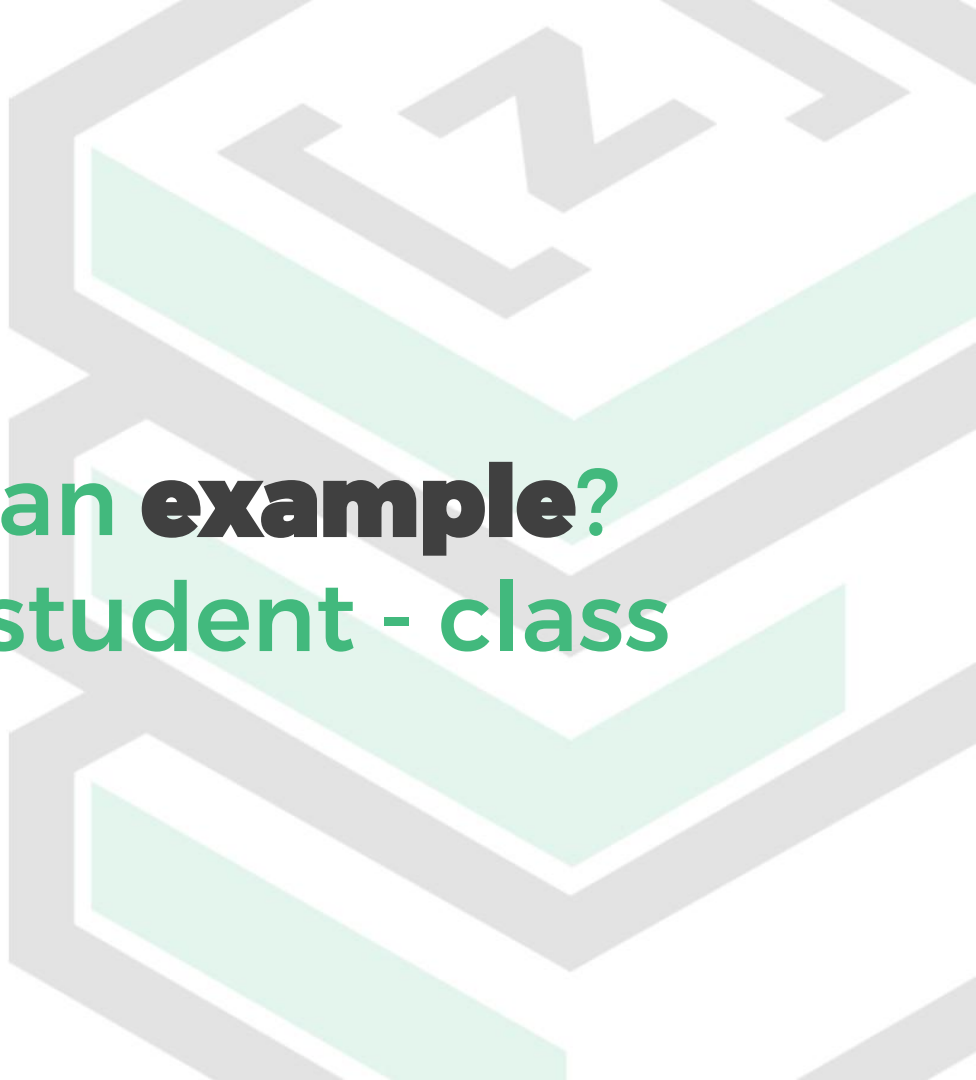
In general, you don't have to do anything with derived relationships.

Cardinality

Cardinality defines the **numbers** at each end of a relationship.

---

e.g., 1:N, 1:1, N:1, N:M



Shall we try an **example**?
e.g., faculty - student - class

Why should we store the **foreign key** to another table rather than the data directly?

Shall we try another **example**?
e.g., meetup group - members

Many to many

‘Many to many’ relationships **can not directly** be implemented in relational databases

---

We must break down the ‘many to many’ relationships into 2 ‘one to many’ and use an **intersecting entity**.

Shall we try yet another **example**?
e.g., patient - medication

RELATIONAL DATABASES PART III

Anomalies? Inconsistencies? Redundancy?

Normalisation is the **process** of improving the design of a database in order to achieve its **optimum** structure

Anomalies? Inconsistencies? Redundancy?

Normalisation tries to eliminate **redundancy**, improve **data integrity** and remove **inconsistencies**

Anomalies? Inconsistencies? Redundancy?

Normalisation tries to rid databases of **update**, **insertion** and **deletion** anomalies

1st Normal Form

Table has only data related to a single entity, no repeating attributes, has only **atomic values** and has a primary key

---

Querying and manipulating data within a table which is not normalised involves more **complexity** than is really needed.

1st Normal Form

PRODUCT

PRODUCT	COLOUR	PRICE
1	red, green	15.99
2	yellow	23.99
3	green	17.50
4	yellow, blue	9.99
5	red	29.99

PRODUCT PRICE

PRODUCT	PRICE
1	15.99
2	23.99
3	17.50
4	9.99
5	29.99

PRODUCT COLOUR

PRODUCT	COLOUR
1	red
1	green
2	yellow
3	green
4	yellow
4	blue
5	red

The PRODUCT table is not in first normal form because the [COLOUR] column can contain multiple values. For example, the first row includes values "red" and "green."

2nd Normal Form

1NF + all non-PK attributes are fully **functionally dependent** on (the whole) of (every) PK

---

e.g., student_email in student_class table violates 2NF

2nd Normal Form

PURCHASE DETAIL

CUSTOMER ID	STORE ID	PURCHASE LOCATION
1	1	Los Angeles
1	3	San Francisco
2	1	Los Angeles
3	2	New York
4	3	San Francisco

PURCHASE

CUSTOMER ID	STORE ID
1	1
1	3
2	1
3	2
4	3

STORE

STORE ID	PURCHASE LOCATION
1	Los Angeles
2	San Francisco
3	New York

The PURCHASE DETAIL table has a composite primary key [CUSTOMER ID, STORE ID]. The non-key attribute is [PURCHASE LOCATION]. In this case, [PURCHASE LOCATION] only depends on [STORE ID], which is only part of the primary key. Therefore, this table does not satisfy 2NF.

3rd Normal Form

2NF + Every non-prime attribute is **non-transitively dependent** on every key of the table

---

The only data that we store from another table should be just enough to create the relation. Otherwise it is redundant data. We can use the id to the other table and find all the info.

3rd Normal Form

BOOK DETAIL

BOOK ID	GENRE ID	GENRE TYPE	PRICE
1	1	Gardening	25.99
2	2	Sports	14.99
3	1	Gardening	10.00
4	3	Travel	12.99
5	2	Sports	17.99

BOOK

BOOK ID	GENRE ID	PRICE
1	1	25.99
2	2	14.99
3	1	10.00
4	3	12.99
5	2	17.99

GENRE

GENRE ID	GENRE TYPE
1	Gardening
2	Sports
3	Travel

In the BOOK DETAIL table [BOOK ID] determines [GENRE ID], and [GENRE ID] determines [GENRE TYPE]. Therefore, [BOOK ID] determines [GENRE TYPE] via [GENRE ID] and we have transitive functional dependency, and this structure does not satisfy 3NF.

LAB 2

Within your team

- Work in pairs on your project
- Identify the entities (look for nouns)
- Identify the relationships between the entities
- Transform ‘many to many’ relationships using intersecting entities
- Identify the attributes for each entity and normalise those entities to 3NF
- Identify primary keys
- As a team outline on paper your database design.

Shall we **share?**

Within your team

- Is there something you can improve in your design?
- Why?

SOME TIPS?
(FROM BITTER EXPERIENCE)

Where would you start your **data modelling** from?

Why would you choose a **relational** database?

How **powerful**
(flexible, fast, accurate)
do you want to be at storing,
retrieving and producing
information?

This material was based on

Mike Zellers - CISS143 Database Design course

https://www.youtube.com/watch?v=ZFpRzr8hCm4&index=12&list=PL38oYrcNWuOOUcLvQ_a7EEu8-mLqVfQZg

---

Bitter experience

and http://www.1keydata.com/database-normalization/


“The goal of Ruby
is to make
programmers happy”
- Yukihiro Matsumoto


WE LOVE SHARING INSIGHTS
JOIN OUR NEWSLETTER
zanshinlabs.io
thank you