

most 'Tables' you work with in an organization are actually views

When you create a view
you can prefix it with VIEW
or even VN -

For tables you can prefix it
TBL -

You can create privileges to
restrict what users can see

DESIGNING DATABASES

apply data modelling
tech → DB design

Normalize

restructuring data b
defined normal forms

eliminates data redundancy
+ inconsistencies

3 Common forms

FIRST NORMAL FORM 1NF

2NF

3NF

1NF

EACH FIELD IN A TABLE
SHOULD CONTAIN A
SINGLE VALUE

EACH ROW IS UNIQUE

Rows can have a
repeat field

But whole rows don't match

family
Smiths
Jones

children
Chris Abby Susan
Steve Mary Dillon

family

Smiths
Smiths
Smiths
Brooks
Brooks
Jones

Chris
Abby
Susan
Steve
Mary
Dillon

2NF

BE IN INF.
SINGLE COLUMN PK
PK

IDENTIFIES ROWS UNIQUELY
GENERALLY THERE COULD
BE A NEED TO CREATE A
NEW TABLE

Smiths A
Smiths
Smiths
Jones
Jones
Jones

TRAT
T16
COW
COW
COW
TRAP

family Table

1 Smith
2 Jones

DEF

ON
C
AC

Child Table

Child ID family ID Children

11	1	
22	1	
33	1	
44	2	
55	2	
66	2	

SET
S

A
S

O
C

S

TRANSITIVE DEPENDENCE IS
THE RELIANCE OF A COLUMN
ON ANOTHER COLUMN THROUGH A THIRD
COLUMN

TRANSITIVE

IF $X > Y$ AND $Y > Z$
THEN $X > Z$

DEPENDENCE

ONE VALUE RELIES ON ANOTHER
CITY RELIES ON ZIP
AGE DEPENDS ON BIRTHDATE

Say you have 3 cols
StoreName OwnerName OwnerAdd

Owner Name + OwnerAdd DEPEND ON
StoreName

OwnerAdd also relies on the
Owner Name

So OwnerAdd relies on
StoreName via OwnerName

~~3NF~~
IN 2NF

CONTAIN NON TRANSITIVE
DEPENDENCIES

PET NORMALIZER

No one table should exist
in isolation from all other
Tables

Always have a relationship

FOREIGN KEYS

Always make a table
have a primary key.

This is how you "talk to
it" - PRIMARY KEYS
ARE UNIQUE

e.g. in pets

owner has ID PK
pet has ID PK

POSITIONAL
CONSTRAINTS

ST
OTHER

SHIPMENT

KB
S

foreign key references
key in another table
REFERENTIAL INTEGRITY.

we can tie the table together
using FK

(FK does not have to
(reference PK in other table)
But it must be UNIQUE)

FOREIGN KEY (columnname)
REFERENCES
table-name (colname)

If there is a FK and you
insert values then the
fk has to exist in the
table the FK refers to.

You will get an error that says
violates FK constraints

key (colname) = (val) is not
present in table table name

Because of REFERENTIAL INTEGRITY

It will not let you add if it
does not exist.

That way you eliminate
REDUNDANCY.

Eg no child will exist without
a parent
no pet without an owner

Also it will not let you
UPDATE things ~~unless~~ because
they exist

You can't delete the parent
You have to remove the
child

- you have to delete
them in order

THAT WAY YOU MAINTAIN
DATA INTEGRITY (REFERENTIAL
INTEGRITY)

DATA MODELING

Foreign Keys reference the
PK of another table
can have a diff name
need not be unique

Relationships

one To one

one To many

many To many

1-1 ONE TO ONE

each item in one col
is linked to only one
item from another col

EG SSN - PERSON

1

1

ONE TO MANY

ONE ADDRESS TO MANY PPL
Each person has an address

So if some family moves
you just have to change
the address once in the
master address table

MANY TO MANY

ID Child
1 Bart
2 Lisa
3 Maggie

ID Parent
11 Homer
12 Marge

EACH CHILD CAN HAVE
MORE THAN ONE PARENT

EACH PARENT CAN HAVE
MORE THAN ONE CHILD

In a many to many
relationship

The 2 Tables are joined in a
JUNCTION TABLE

Junction TABLE

parent_id

11
11
11
12
12
12

child_id

1
2
3
1
2
3

Join the child and parent table to junction table

Homer
Homer
Homer
Marge
Marge
Marge

Bart
Lisa
Maggie
Bart
Lisa
Maggie

CREATE TABLE child_parent
child_id INTEGER NOT NULL
FOREIGN KEY (child_id) REFERENCES
children (child_id)
parent_id INTEGER NOT NULL
FOREIGN KEY (parent_id) REFERENCES
parents (parent_id)
PRIMARY KEY (child_id, parent_id)

JUNCTION TABLE

SELECT children.child-name,
child-parent.child-id,
parents.parent-name,
parents.parent-id

FROM children
LEFT JOIN child-parent
ON

child-parent.child-id =
children.child-id

LEFT JOIN parents
ON

child-parent.parent-id =
parent.parent-id;

INNER JOIN

When you join you drop
records that don't match

OUTER JOIN

Matches everything

Left

Keeps integrity of leftmost
table

Anything that does not
match gets dropped

VENN Diagram

ENTITY RELATIONSHIP DIAGRAM

a visual representation
of entity relationships
within a database

How are entities related

- Primary key
- Foreign key
- one to many
- one to one
- many to many

Before you create the DB
(tables) you map out all
the relationships in a
diagram

Conceptually draw a diagram

Iterate and improve on
diagram that represents your
DB before you create it

You can also create a view
diagram to show all
relationships

Entites
Their data types
and relationships
are all illustrated in ERD

There are 3 models used
when creating ERD

Conceptual

Basic info containing
tbl names + col

→ if you
so lin

→ relate

→ link

Logical

Slightly more complex
with IDs and datatypes

So
Train
of C

Physical

The BLUEPRINT of your DB

REFLECTS PHYSICAL
RELATIONSHIPS BETWEEN
ENTITIES

CPU
mem
refe

In que

Pay

Gym ex
conce
logica

Gym example

conceptual.txt

logical.txt

→ if you don't pay, you can't go
so link member to payments

→ relate trainer to members

→ link trainer to a gym

So create a FK in
Trainer that links to PK
of Gym

You can't make payment if not Member
may add FK member ID
reference payment PK

In quickdatabase diagram

PaymentId INT FK-Payments.ID

- is for one to one

Look Under does together
full list

UNION (optional)

one liner.

Toys

UNION

complete list
from 2 tables

```
Select toy_id AS  
      id, type  
From toys
```

UNION

```
Select game_id AS id, type  
From games
```

If you want to include
duplicates you say
UNION ALL

Just generates a list

next adu data storage +
retrieval