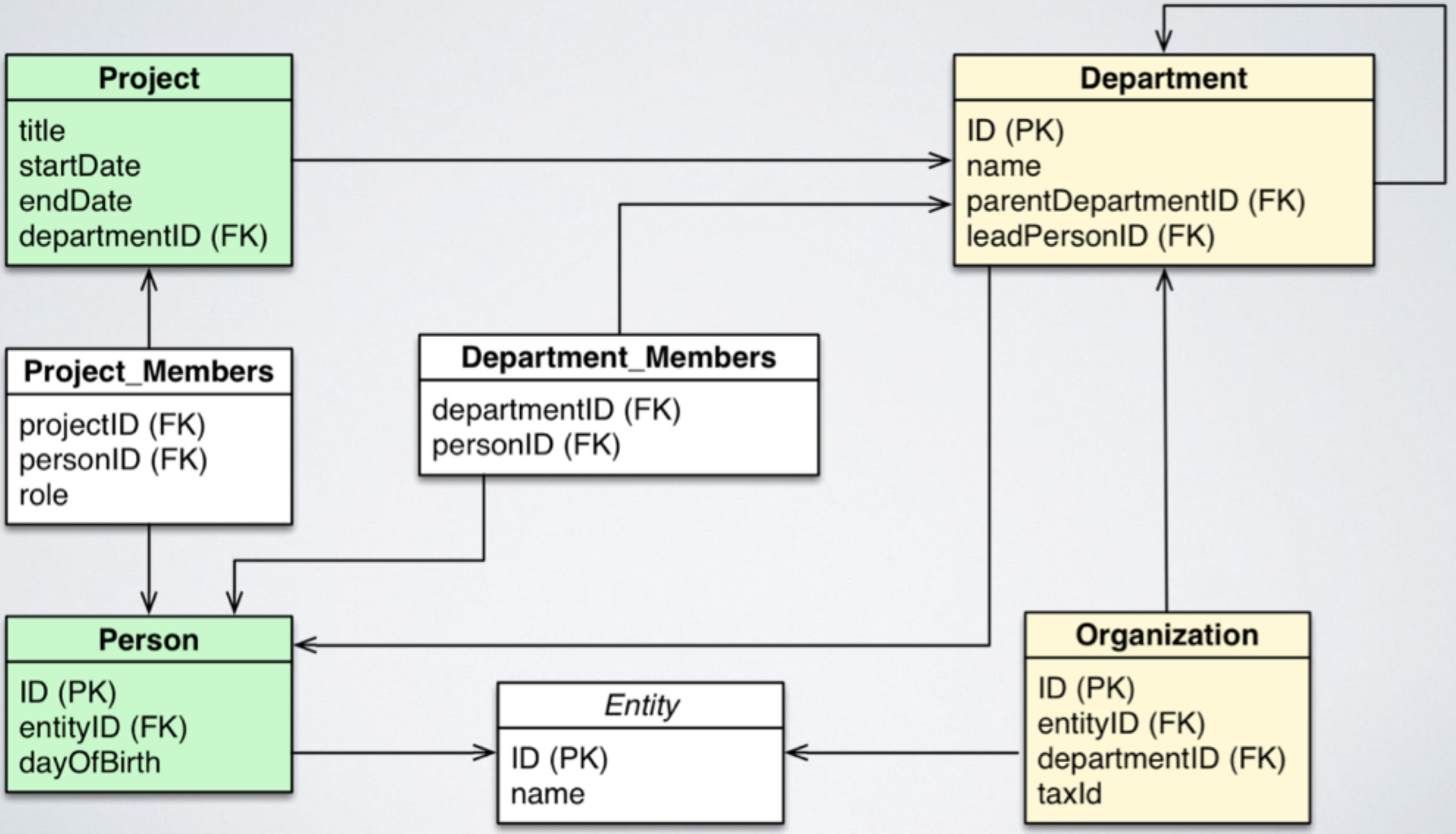
ACTIVERECORD & DATABASE

D-Labs
Jan Berdajs
April, 2017

DATABASE TYPES

- Document-orientated
- Graph
- Relational
  - SQL
- Many more…

ENTITY-RELATIONSHIP MODEL



RAILS - ACTIVERECORD

- ORM - Object-relational mapping
- Database abstraction that makes working with the database fit better with object-orientated environment
- Model classes - ActiveRecord pattern: kind of fat model (persistence and domain logic is not separated)
- Converts database table-style data into Ruby objects and vice-versa

SQL

- Structured Query Language
- Databases: SQLite, PostgreSQL, MySQL, MsSQL, …
- Each database implementation may have non-standard SQL, especially for advanced features

CREATING THE MODELS

- rails g model Category title slug
- rails g model Post title content slug published:boolean published_at:datetime category:references
- rails g model Comment name content post:references approved:boolean
- rake db:migrate db:seed
- app/models, db/migrate, db/schema.rb

TOOLS

- rails console

- rails dbconsole

QUERYING

- Post.all
- Post.where(published: true).first
- Post.where(title: "Post 1'").first
- Post.where.not(title: "Post 1").first
- Post.where("published_at > ", 10.days.ago).first

QUERYING

- Post.where(title: ["Post 1", "Post 2"])

- Post.where("title LIKE ?", "%Unpublished%")

ORDERING

- Post.order("title")
- Post.order("title asc")
- Post.order("title desc")

JOINS

- Post.joins(:category).where("categories.title = ?", "General")

INCLUDES

- What is includes?
- Post.first(10).map(&:category)
- Post.includes(:category).first(10).map(&:category)
- Post.includes(:category).joins(:category).where(categories: { title: "General" })
- Post.includes(:category).references(:category).where(categories: { title: "General" })

DEMO PROJECT

- https://github.com/mrbrdo/ar-workshop-apr-2017
- Implement filter_posts in posts_controller.rb
- Do not apply filtering before the filter form is submitted (hint: filter=on URL parameter)
- Only show posts in selected categories, or all if no category is checked
- Filter only published posts, filter by published time
- Filter by post title using LIKE, so all posts that contain the text in the title are matched. Don't filter if input is left empty.
- Order posts based on what is selected in the form (you can modify the form if you want). For ordering by category title you will need to use joins.
- Since we are displaying category title for every post, think how you could optimize the performance by reducing the number of queries (check rails server log to see the queries).
- Implement show action (find post by slug which is in params[:id])

HOMEWORK

- add some comments in seed.rb
- add a filter to only show posts that have comments posted in certain time (from-to) - use joins
- **advanced 1 (optional):** add a filter by number of comments (you can have only 1 input to enter the exact number) - hint: use "group" (SQL GROUP BY), "having" and "count", or try to find out how using stackoverflow/google. Looping over each post in Ruby code and checking the number of comments for each is not the allowed solution.
- **advanced 2 (optional):** add paging to results (show 5 results per page). the paging should work with filtering (on pages after the first too). hint: use "limit" and "offset"