* Cần dẫn dắt để đi từ thủ tục và oop,
* Ví dụ chưa điểm hình,
* Có thể tham khảo các ví dụ kinh điểm,
* Xử lý tình huống chưa tốt, nên gt code có sẵn trc,
* Nên demo theo tạo class,
* Cần nêu được sự khác biệt rõ giữa oop và pop,
* Cần cho các bạn hiểu rõ về class và object,
* Ko nên quá dựa slide,
* Chưa có ví dụ,
* Nên demo ngay khi nêu khái niệm của lớp rồi …,
* Nên có ví dụ xuyên suốt,
* Hơi bám lý thuyết,
* Nên mô tả phân biệt với các kiến thức cũ,
* Nên dùng slide cũ của codegym.

Chị Châu:

* Cần thông được tư duy,khả năng hệ thống của học viên
* Liên kết bài học với hệ thống, vị trí bài học trong lộ trình
* Mở đầu nên đứng ở trung tâm
* Chưa dẫn dắt về OOP
* Học viên đưa ra mục tiêu
* Cần chốt lại sau từng mục tiêu
* Định hướng cho những phần khó
* Demo: viết cú pháp lên bảng, tại sao cần sơ đồ lớp, giúp học viên có cách tư duy,

Bảng trello của lớp.

Báo cáo tuần.

Báo cáo khoá.

Ghi nhật ký Retros và andy và dẫn link cải tiến.

Email báo cáo kết thúc module.

Phiếu dự giờ giảng viên.

Set-up thanh chrome.

Đào Tạo\_CGMon, 1. Phòng đào tạo, Báo cáo: Tuần, kết thúc module, Báo cáo kết thúc chương trình (Biểu mẫu: Quy trình => Quy trình Đào Tạo1.6 =>), Retros hàng tuần (C12), Đánh giá năng lực,

// SQL

* Làm rõ vai trò của sql

1. Thu thập: chỉ biết, coi như đã khảo sát (đặt tả yêu cầu).
2. Phân tích: thu được ERD (Mô hình thực thể quan hệ: Entity relationship diagram).
3. Thiết kế: thu được table diagram.
4. Triển khai: truy vấn

* Sau 2 bài sql phải thiết kế được một csdlqh cơ bản
* Định hướng: 2 bước đầu là khó và chưa tiếp cận được, phải nắm được kiến thức cơ bản.
* Bài 1: làm rõ các khái niệm
  + Giới thiệu trước lộ trình
  + RDBMS: Cơ sở dữ liệu quan hệ, xử nhỏ từng chữ viết tắt
    - Relational Database Management System

1. Data (hỏi học viên):
   1. Dữ liệu là gì? Là tất cả những thứ ta thu thập
   2. Nếu thế thì information và data khác nhau ở đâu? Xử lý dữ liệu ra được thông tin
   3. Data là dữ liệu thô(input), information là dữ liệu đã được xử lý(output)
   4. Có thể là thông tin trong ngữ cảnh này, dữ liệu trong ngữ cảnh khác
2. Database:
   1. Tổ chức dữ liệu hỗ trợ truy xuất sau này
   2. Ai có thể lấy ví dụ csdl thực tế??? (Cuốn sách…)
   3. Người lập trình quản trị csdl
3. Database management:
   1. Tổ chức
   2. Thao tác
4. Database Management System: => nên focus Relational Database Management System (phổ biến nhất).
5. Relational Database:
6. Relational Database Management:
7. Relational Database Management System:
   1. Mô hình quan hệ
      1. Bảng: các hàng(bản ghi), các cột(trường, thuộc tính)
      2. Khoá chính: để xác định duy nhất 1 hàng, 1 thực thể trong bảng
      3. Mối quan hệ
      4. Khoá ngoại

* Bài 2: Phân tích thiết kế:
  + ERD:

1. Entity: Lấy ví dụ: KHA mua SB> thực thể KHA, SB
2. Relationship: mua
3. Properties/ Att: KHA: tên A, ngày sinh …, giới tính …; Sách: tên: B, bìa: …, số trang: …. => xây dựng bài toán tổng quát, cơ chế: các thực thể có các thuộc tính, đặc trưng như nhau (KH và S)
4. Entity Set: KH và S là các entity set, tuỳ mỗi bài toán mà thực thể nằm ở trong nhóm khác nhau
5. Relationship Set: quan hệ của các entity set
   * + - Nhấn mạnh: từ loại là gì? Entity danh từ chung, đối tượng là danh từ riêng, mqh là động từ.
   * Phân loại mối quan hệ: phân loại đi kèm tiêu chí
     + Số lượng thực thể tham gia vào mqh:
       - Tự thân: mqh xảy ra trong tập thực thể đó.
       - **Nhị phân: xảy ra giữa 2 thực thể giữa 2 tập thực thể khác nhau**
       - Tam phân: (Ngân hàng, Nhân viên, Chi nhánh, Phòng ban: NV a làm việc ở phòng KT ở Mỹ Đình)
     + Ánh xạ: bài toán tập hợp X và Y
       - 1-1:
       - 1-n:
       - N-1:
       - N-m:
         * Ví dụ: …
         * Bài toán xuyên suốt: KH, HĐ, Sách. Mỗi sách nhiều hoá đơn là tập thực thể!
         * Vẽ ERD: có mqh, hành động, thuộc tính và khoá chính. Do hệ csdl sẽ có nhiêu bảng nên trong thực tế sẽ luộc bớt thuộc tính và tách riêng.
         * Mqh n-m là khó nhất: “thuộc tính của mối quan hệ” (Giá và SL cùng có trong HĐ và Sách => Giá của sách và giá tại thời điểm bán, tại sao có 2 nơi lưu trữ ‘giá’? => thời điểm sau đó bán giá khác nên cần có giá trong HĐ để lưu lại giá tại thời điểm bán, Số lượng sách bán ra và số lượng sp hiện có trong kho).

* ERD không có khoá ngoại.
* **2 MQH nhiều nhiều điển hình: KH-Bình luận-Sách và HĐ-SP**
* **ERD => Table design???**
  + Quy tắc:

1. Mỗi thực thể được chuyển thành 1 bảng:
   1. Trường: các thuộc tính
   2. Khoá chính: Khoá chính
2. Xét mối quan hệ:
   1. 1-n: bổ sung khoá chính của bảng 1 vào làm khoá ngoại của bảng nhiều
   2. 1-1: bổ sung khoá chính của bên này làm khoá ngoại của bên kia
   3. N-m: sinh bảng trung gian, khoá chính của 2 bảng chuyển thành khoá ngoại ở bảng trung gian và các thuộc tính. Khoá chính của bảng trung gian: nếu cặp khoá ngoại đủ điều kiện (not null + unique) hoặc sinh ra khoá chính.

* SQL: Structral query language
  + DDL: data definition language: create alter drop
    1. DB
    2. Table: nói về rằng buộc trước.
    3. Rằng buộc: constraint: notnull, unique, primary key, foreign key, default, check (sau check là điều kiện: so sánh, in, and or, not in, between and, like, …)
  + DML: data manipulation language: select insert update delete
    1. Mệnh đề: from, where, group by, having, order by, limit,
       1. Select + \*/ds cột/ hàm tập hợp(max, min, **count**, avg…) => liệt kê hàm tập hợp => so sánh một số hàm tập hợp.
       2. From + bảng/ join
       3. Where + biểu thức điều kiện
       4. Group by: nhóm các bộ kq
       5. Having: so sánh having (hàm tập hợp) và where (cột), mqh having (tuỳ chọn đi kèm group by) và group by.
  + DCL: data control language: grand deny revoke

**Truy vấn lồng, join: vd (hiển thị dssv chưa thi môn nào).**

* Truy vấn lồng: trong truy vấn có chứa 1 truy vấn. (lấy danh sách sv chưa thi môn nào).
* Join: lấy dữ liệu trên nhiều bản. (Bài 4)

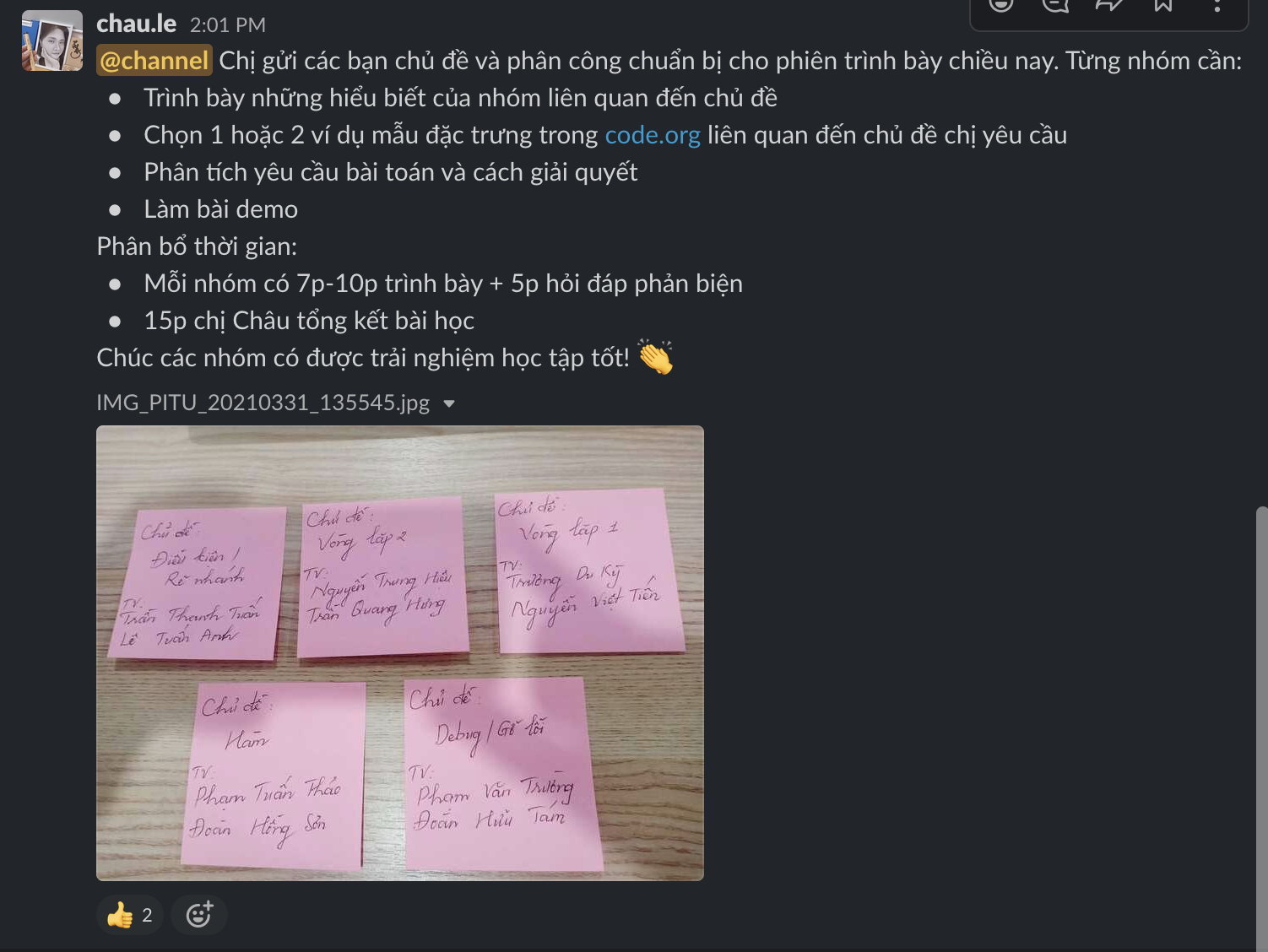
1. *Tích decac: nhân chéo: Select \* from b1, b2*
2. Inner join(default): giới hạn bớt số bộ từ tích decac (chỉ những bộ thoả mã điều kiện).
   * Select \* from b1 join b2 on dk1
   * Select \* from b1, b2 where dk1
3. Outer join: bổ sung null vào các vị trí thiếu (left – thêm null bên phải, right- thêm null bên trái, full outer – bổ hết các bộ chung).

* Các đối tượng nâng cao: view, index, procedure, function, trigger
  + Index: đánh chỉ mục, tăng tốc độ truy xuất. ý nghĩa, khi nào (nên đánh trên các cột có tính phân biệt cao, dựa vào mục đích, dữ liệu có tính ổn định).
  + *Hệ số điền đầy.*
  + View(bảng ảo): là kết quả của truy vấn select, các query lặp lại nhiều lần, khi muốn lấy dữ liệu trên nhiều bảng. \*(temporary table)
  + **Procedure: hàm, vào ra, vừa vào vừa ra**
  + *Trigger: bẫy lỗi, thay vì.*

**Buổi 2: CodeOrg**

**Hàm:**

Xác định đầu vào, đầu ra

****

**Cộng điểm:**

* + Hàng ngày viết post cộng điểm + khen ngợi
  + Các bạn làm tốt, chủ động chia sẻ với lớp, giúp đỡ bạn học.
* Chữa bài theo từng dạng:
  + Giai thừa
  + Hiển thị 20 số … đầu tiên
* Mảng: lưu trữ nhiều giá trị.
  + Nhập xuất => Tìm kiếms => Tìm min, max =>Sắp sếp=>đảo ngược => trộn
  + Cách truy xuất, dùng for

Mảng:

* Mảng lưu trữ liên tiếp trong bộ nhớ.
* Mỗi ptu tương đương 1 biến.
* Chỉ số: 0->n-1
* Phần tử: tên[chỉ số]

Mảng 2 chiều: mảng của các phần tử, mỗi phần tử là một mảng ⬄ Mảng của mảng.

* Ví dụ
* Cách truy xuất phần tử
* Cách duyệt
* Vẽ bảng

**Bài Hàm:**

* Liên hệ hàm ở Codegorg
* Tỉnh tổng lặp đi lặp lại.
  + - * Giúp code ngắn hơn.
      * Đọc hiểu code dễ hơn
      * Sửa dễ hơn.
* Cách viết được hàm:
  + Cách viết: cú pháp, quy tắc đặt tên.
  + Input output: xác định, input được đẩy vào làm tham số, output là return.
  + Các bài tập… đầu voà đầu ra
  + Gọi hàm, cú pháp gọi hàm
* MindMap, Làm bộ câu hỏi =< drive
* Đánh giá năng lực học viên sơ sài
* CV học viên giống nhau, sơ sài
* Học viên PV nhắc bài nhau