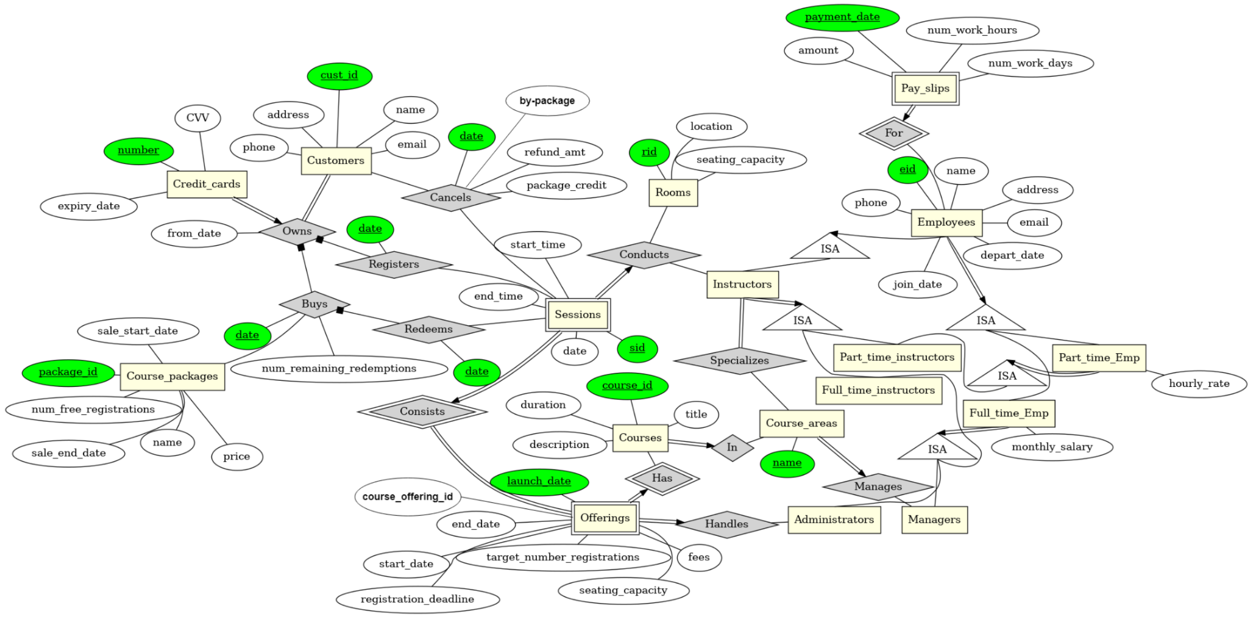
Project Team 114

Name	Student Number
Tan Wei Da Jeremy	A0183861X
Wong Peng Yu	A0200407Y
Wong Tau Yew Mark	A0190753B
Yang Yang	A0168815U
Yew Kai Zhe	A0191094A

Project Responsibilities

Name	Responsibilities
Tan Wei Da Jeremy	Design and create ER Diagram, create functions, alter schema when appropriate, test functions
Wong Peng Yu	Design and create ER Diagram, create functions
Wong Tau Yew Mark	Design and create ER Diagram, create functions, create schema, alter schema, test functions, create data.sql
Yang Yang	Design ER Diagram
Yew Kai Zhe	Create functions

ER Data Model



Design Decisions

We kept as close to the given ER model as possible because the given ER model most likely at least guarantees a possible correct answer. If we implemented our own ER model, we may have overlooked an important detail, which would make getting the correct answer impossible.

Non-Trivial Design Decisions

We added a course_offering_id attribute to the course_offering schema because several functions required it as input. Thus, it was impossible to not implement it unless the input was changed from course_offering_id to course_id and launch_date.

A by_package attribute is added to the Cancels entity to help in differentiating whether a cancellation is made for a redemption or registration by credit card especially when there is late cancellation where both package_credit and refund_amt are 0. This attribute aids in the get_my_course_package functionality to retrieve the redeemed sessions correctly since late cancellation of registrations via credit cards and late cancellation of redemptions can be easily differentiated. Another way is to divide the Cancels into 2 sub-tables, one that maintains the cancellations of registrations via credit cards and another that maintains the cancellations of redemptions. This has the benefit of expressing appropriate attributes as foreign keys to the Registers or Redeems table. However, the former way is implemented for conciseness.

A cancellation does not remove the corresponding entry in the Redeems table or Registers table even if the cancellation results in a refund. This is to preserve the history (dates) of when the registrations (by credit card or redemption) are made. This also ultimately affects the promote_courses functionality that the course areas associated with the refunded sessions may better reflect the interest of customers than those that customers already successfully registered and completed.

To limit exposure of information from customers, the update_credit_card functionality changes the credit card details of an existing customer instead of adding a new credit card entry. This change in turns affects many tables that depend (directly or indirectly) on the credit card number via a foreign key. As such, ‘on update cascade’ is added to these tables on these foreign keys to uphold the database. However, it is noted that this is a limitation as well since old purchases of course packages, registration via credit card and cancellations made cannot be directly traced back to the exact credit card details used for payment. However, the customer can still be reached should this information be needed.

Database Schema Constraints Not Captured

1. Each part-time instructor must not teach more than 30 hours for each month.
2. Seating capacity of a course offering is equal to the sum of the seating capacities of its sessions.
3. A trigger (Add_appropriate_sid) is needed to enforce sid to start from 1 for each course offering instead of serial.
4. For a credit card payment, the company’s cancellation policy will refund 90% of the paid fees for a registered course if the cancellation is made at least 7 days before the day of the registered session; otherwise, there will be no refund for a late cancellation. For a redeemed course session, the company’s cancellation policy will credit an extra course session to the customer’s course package if the cancellation is made at least 7 days before the day of the registered session; otherwise, there will be no refund for a late cancellation.
5. For each course offered by the company, a customer can register for at most one of its sessions before its registration deadline.

Triggers

Add_appropriate_sid

This is to ensure the sessions for a course offering are numbered consecutively starting from 1. If there are gap/s in the session identifier (sid), inputting sid=0 in an insertion into the Sessions table also allows for the trigger to find an appropriate sid that can fill in

the gap/s. Gap/s may result after the add_session function is called where the sid specified is arbitrary. This works towards the consecutive numbering of the sessions within each offering.
Alternatively, this numbering can be implemented in the functions, specifically in the add_course_offering and add_session functions. However, for generalisability purposes, implementing it as a trigger is more suitable.

Insert_payslip_once

This is to ensure that any insertion into Pay_slips is only carried out at the end of the month(last day of the month). The trigger only allows the insertion when the addition of 1 day changes the month. For example, 31 March is the end of the month. At 10 March, the addition of 1 day does not change the month, so insertions into Pay_slips cannot be carried out. On 31 March, the addition of 1 day makes it 1 April, and the month changes, hence insertions into Pay_slips is allowed on 31 March.

Target_less_than_seating_capacity

This is to ensure the seating capacity of the course offering must be at least equal to the course offering's target number of registrations.
Implementing this as a check table constraint is not possible since the remove_session functionality allows this constraint to be violated. As such, this trigger is implemented with reference to before insertions into the Offerings table which happen only during an add_course_offering call to differentiate from updates on the Offerings table which is allowed to violate the constraint. Again for generalisability, it is better to implement this functionality as a trigger rather than in the add_course_offering function.

Difficulties encountered and lessons Learnt

PostgreSQL server kept crashing and it will affect the schedule.
Working in a team means being on the same page with your teammates with regards to the relational schema. Just because you want to change something doesn't mean you can. Your changes may end up affecting other peoples' functions/tables. This means having to discuss or at least notify your teammates when you want to change the schema. Even after implementing the change, you have to be ready to revert back to the old schema.