Introduction to Project 1

Discussion Session 2

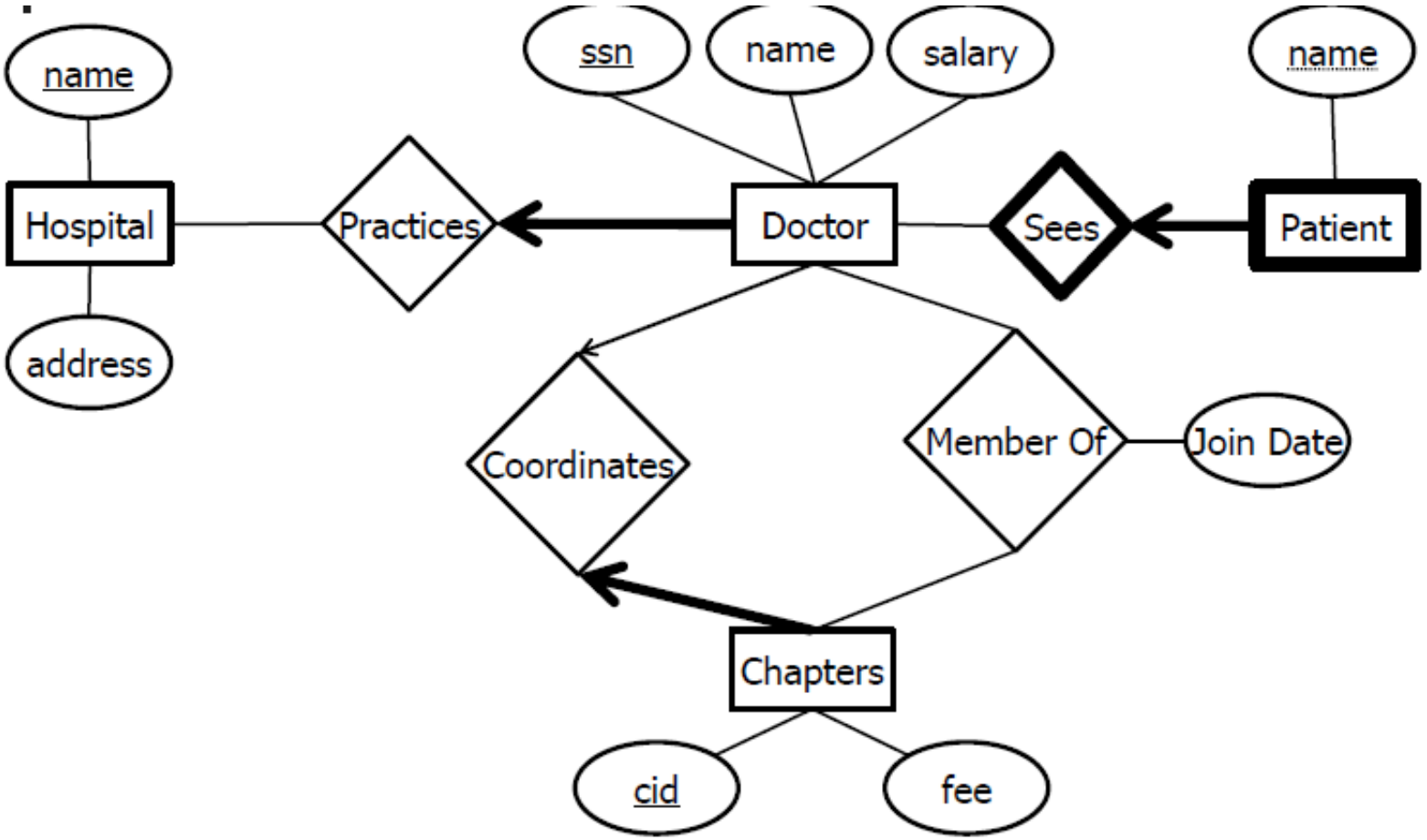
Outline

- ER Design Exercise
- CAEN and Oracle Account Setup
- Project 1 Overview
- Access Fakebook Data
- Using SQL*Plus

ER Diagram Exercise 1

- There are doctors, and each doctor has a name, a salary, and a unique ssn. There are hospitals, and each hospital has an address and a unique name. Each doctor practices at exactly one hospital.
- There are patients, and each patient has a name. Each patient MUST be associated with exactly one doctor, and no two patients of a given doctor have the same name (though two patients of the different doctors can have the same name). In the database, patient tuples should be automatically deleted if the corresponding doctor tuple is deleted.
- There are chapters of the American Medical Association, and each chapter has a unique cid and a membership fee. Doctors can be members of zero or more chapters; it is important to maintain the date on which a doctor joined a chapter. Each chapter has a coordinator, and only doctors can serve as chapter coordinators. No doctor can be coordinator for more than one chapter.

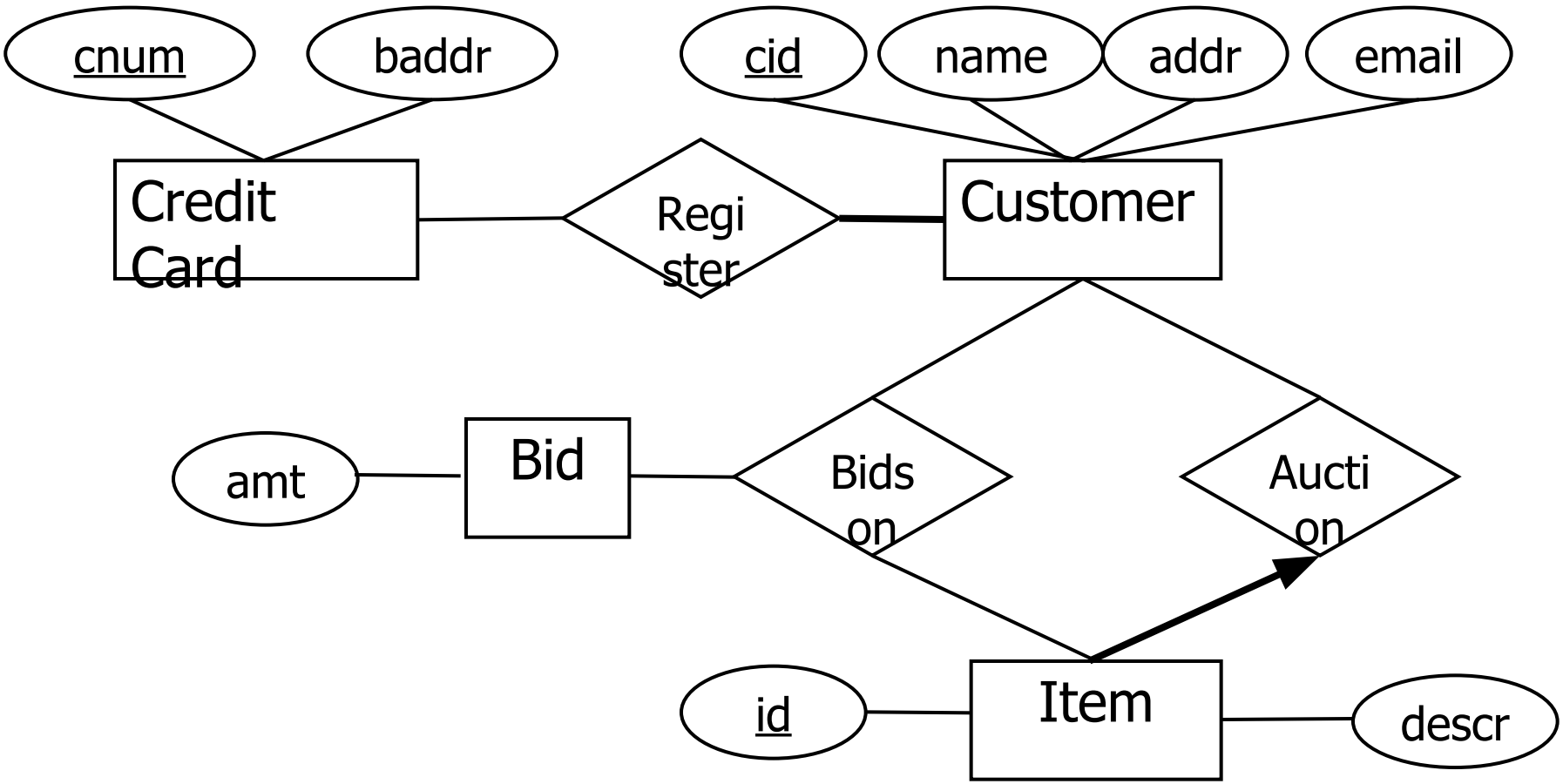
Exercise 1 Answer



ER Diagram Exercise 2

- You have been hired to design a database for a company that runs online auctions.
- Customers register to use the auction site. Each customer has a unique customer ID, a name, a mailing address, and an e-mail address. Customers have the option of registering one or more credit cards with the site. For each credit card, the site must store the credit card number, as well as the billing address.
- Customers can list items for auctions on the site. Each item has a description and a unique item ID. Each item is listed exactly once.
- Customers can also bid on items. Each bid has an associated amount. *Since this is an auction site, it is important that the same customer be allowed to bid more than once on the same item!*

Exercise 2 Solution



CAEN and Oracle Account Setup

- Do you have a CAEN account?
  - If you are not a student in CoE, you do not have a CAEN account by default. If you do not have a account, you'll have to go to the CAEN Hotline(http://caen.engin.umich.edu/hotline) in person with your Mcard to request an account
- Do you have an Oracle account?
  - Username: Your uniqname
  - Initial password: **eecsclass**
  - Sqlplus will prompt you to change password when you log in the first time
  - DO NOT USE QUOTES OR @ IN YOUR NEW PASSWORD. It won't work, and you'll have to talk to one of the instructional staff to reset it.

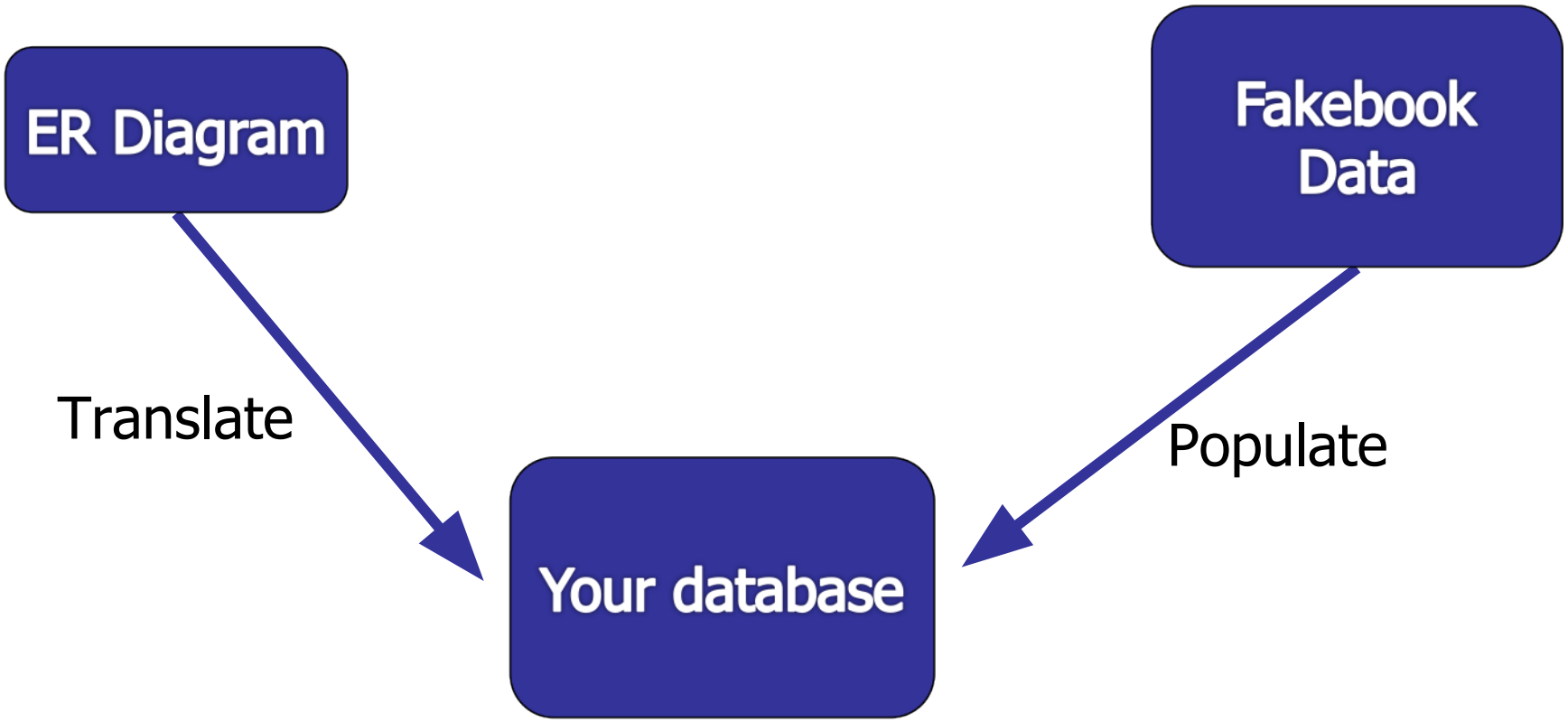
Accessing Oracle Account

- Login (ssh) to CAEN
  - `ssh uniqname@login.engin.umich.edu`
- Setup SQL*Plus Environment
  - `module load oracle/muscle`
    - You must do this every time you log in. Note that you can add this line to your .bash_profile so you don't have to remember it.
- Login to SQL*Plus
  - `rlwrap sqlplus <username>/<password>`
  - Initial password: **eecsclass**

Project 1 Overview

- Goal
  - Design a database using Fakebook data
- Your task
  - Part 1: Create ER Diagrams
  - Part 2: Translate ER Diagrams into relational tables
  - Part 3: Tutorial on how to access public data
  - Part 4: Populate your database

Project 1 Overview



Project 1 Overview

- FAKEBOOK DATA
  - PUBLIC_USER_INFORMATION
  - PUBLIC_ARE_FRIENDS
  - PUBLIC_PHOTO_INFORMATION
  - PUBLIC_TAG_INFORMATION
  - PUBLIC_EVENT_INFORMATION

Part 1: Create ER Diagram

- Read the descriptions carefully to identify constraints
  - **Example**: "Photo Tag" allows at most ONE tag for each user in a photo
- Make decisions based on common sense
  - **Example**: Should location information be an attribute or a separated entity?

Part 2: Create Relational Schema

- Convert ER diagrams into relational tables
- Write DDL scripts to create and drop tables, triggers and sequences
- Capture as many constraints as possible
  - Primary key
  - Foreign key
  - Not null
  - Triggers and sequences

Part 3: Access Fakebook Data

- SELECT user_id

  FROM shravyak.public_user_information;
- Copy the data to your own account
  - CREATE TABLE new_user AS (SELECT * FROM shravyak.public_user_information);
  - DESC new_user; (Prints Table new_user's schema)
  - SELECT user_id FROM new_user;

Part 4: Populating Your Database

```
CREATE TABLE location (
      loc_id integer,
      city varchar(200), state varchar(200),
      country varchar(200),
      CONSTRAINT location_pk PRIMARY KEY(loc_id)
);


INSERT INTO location (city, state, country)
SELECT DISTINCT hometown_city, hometown_state,
hometown_country FROM new_user
UNION
SELECT DISTINCT current_city, current_state, current_country FROM
new_user
UNION
SELECT DISTINCT event_city, event_state, event_country FROM
shravyak.public_event_information;
```

Triggers and Sequence

```
CREATE SEQUENCE loc_sequence
START WITH 1
INCREMENT BY 1;

CREATE TRIGGER loc_trigger
BEFORE INSERT ON location
FOR EACH ROW
BEGIN
SELECT loc_sequence.NEXTVAL INTO :new.loc_id from dual;
END;
.
RUN;
```

Whenever you insert a row into location, this will automatically set the value of primary key **loc_id** to the next integer in the sequence.

Browse your Data

- SELECT table_name FROM user_tables;
  - Lists all the tables created on **your DB.**
- DESC user_information;
  - Gives you the schema of the table you have created
- SELECT count(*) FROM table_name
  - Returns the number of rows in table_name
- SELECT user_id
  FROM new_user;
- SELECT first_name, last_name
  FROM new_user
  WHERE rownum < 3;

Rownum in Oracle: Basics

- Rownum is one way to limit what data you get back; useful if you only want to look at the first N rows.
- Rownum in incremented only after the current number is assigned to a row.
- SELECT first_name, last_name

  FROM new_user

  WHERE rownum < 3;
  - This returns the top two rows.
- SELECT first_name, last_name

  FROM new_user

  WHERE rownum > 3;
  - Doesn't return anything. No row is ever selected, so rownum is never incremented.

Rownum in Oracle Part 2: Sort

- Rownum also complicates things when combined with sort. If you don't do something special, rownum is assigned before the sort happens.
- Select first_name, last_name
  where rownum <=3
  from new_user
  order by last_name;
  - First 3 rows selected BEFORE order by is applied – not guaranteed to be in order
- Select * from
  (select first_name, last_name
  from new_user
  order by last_name)
  where rownum <=3
- Gets the top three ordered by last name