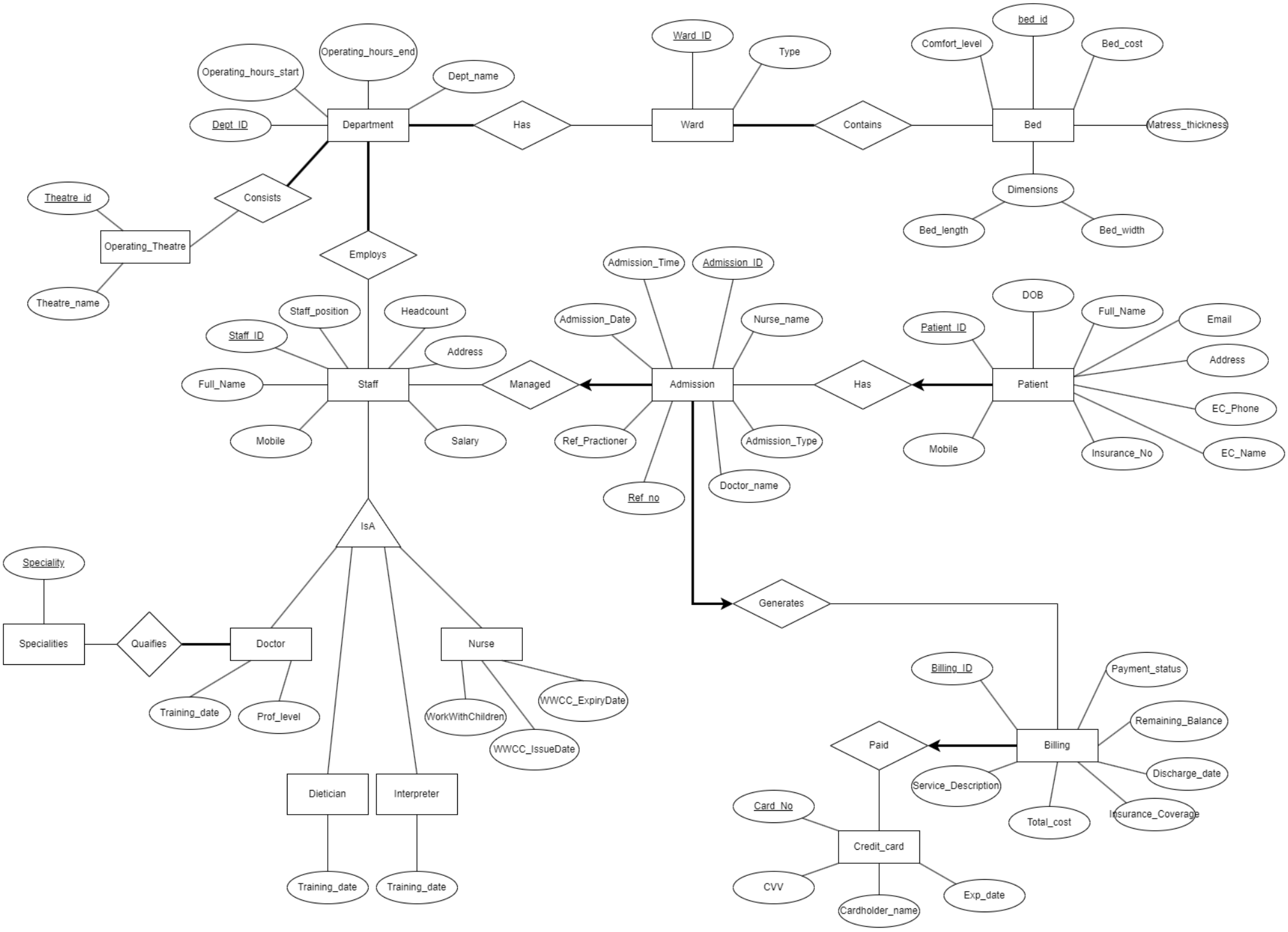
Central Sydney Hospital



ER diagram justification

1. Department:

- Entity type: We have used a strong entity type for ‘Department’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Department’ has three One-to-Many relationships with ‘Ward’, ‘Operating_Theatre’, and ’Staff’ because each instance of department can consist of multiple wards, operating theatres, and staff.
- Attributes: ‘Department’ has got four attributes namely: Dept_ID, Operating_hours_start, Operating_hours_end, Dept_name as per the statement in the assignment.
- Primary keys: The primary key for ‘Department’ is ‘Dept_ID’ as that will not be the same ever for any two tuples in the ‘Department’.
- Constraints: We have used the ‘Check’ constraint to classify the admission based on operating hours and department name, that is whether it is an ‘Emergency Case’ or ‘General Case’.

2. Operation_theatre:

- Entity type: We have used a strong entity type for ‘Operation_theatre’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Operation_theatre’ has a Many-to-One relationship with ‘Department’, because one department can consist of more than one operation theatre.
- Attributes: ‘Operation_theatre’ has got two attributes namely: ‘Theatre_id’ and ‘Theatre_name’.
- Primary keys: The primary key for ‘Operation_theatre’ is ‘Theatre_id ‘as that will not be the same ever for any two tuples in the ‘Operation_theatre’.
- Constraints: We have used the ‘Not Null’ in ‘Theatre_name’ to make sure we do not get any null values for it.

3. Ward:

- Entity type: We have used a strong entity type for ‘Ward’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Ward’ has one One-to-Many relationship with ‘Bed’ because each instance of ward can consist of multiple Beds.
- Attributes: ‘Wardr’ has got two attributes namely: Ward_ID (As it’s the primary key of Ward), and Type as per the statement in the assignment.
- Primary keys: The primary key for ‘Ward’ is ‘Ward_ID’ as that will not be the same ever for any two tuples in the ‘Ward’.
- Constraints: We have used the ‘Not Null’ in ‘Ward_ID’ to make sure we do not get any null values for it.

4. Bed:

- Entity type: We have used a strong entity type for ‘Bed’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Bed’ has a One-to-One relationship with ‘Ward’ because each instance of ‘Bed’ can consist of exactly one instance of ‘Ward’.
- Attributes: ‘Bed’ has got five attributes : Bed_ID(As it’s the primary key of Bed), Bed_cost,Dimensions,Comfort_level and Matress_thickness as per the statement in the assignment.
- Primary keys: The primary key for ‘Bed’ is ‘Bed_ID’ as that will not be the same ever for any two tuples in the ‘Ward’.
- Constraints: We have used the ‘Not Null’ in ‘Bed_ID’ to make sure we do not get any null values for it.

5. Staff:

- Entity type: We have used a strong entity type for ‘Staff’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Staff’ has only one One-to-One relationship with ‘Admission’ because each instance of ‘Staff’ can consist of only one instance of ‘Admission’ as one admission is managed by one staff member.
- Attributes: ‘Staff’ has got seven attributes namely: Staff_ID, Staff_position, headcount, address, salary, mobile and full_name.
- Primary keys: The primary key for ‘Staff’ is ‘Staff_ID’ as that will not be the same ever for any two tuples in the ‘Staff’.
- Constraints: We have used the ‘Not Null’ in ‘Staff_ID’ to make sure we do not get any null values for it.
- Design Speciality: We have created a ISA relation between the ‘Staff’ entity which is the super class and other entities like ‘Doctor’, ‘Nurse’, ‘Dietician’ and ‘Interpreter’ as the sub classes.

6. Dietician:

- Entity type: We have used a strong entity type for ‘Dietician’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Dietician’ is one of the specialisations of the entity ‘Staff’ so, it is One-to-One in relationship with ‘Staff’.
- Attributes: ‘Dietician’ has got two attributes namely: Staff_ID (As it’s the primary key of Staff), and Training_date as per the statement in the assignment.
- Primary keys: The primary key for ‘Dietician’ is ‘Staff_ID’ as that will not be the same ever for any two tuples in the ‘Dietician’.
- Constraints: We have used the ‘Not Null’ in ‘Staff_ID’ to make sure we do not get any null values for it.
- Design Speciality: We have created ‘Dietician’ as the sub class from ‘Staff’ super class using ISA relationship.

7. Doctor:

- Entity type: We have used a strong entity type for ‘Doctor’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Doctor’ is one of the specialisations of the entity ‘Staff’ so, it is One-to-One in relationship with ‘Staff’ and also has a One-to-One relationship with specialities .
- Attributes: ‘Doctor’ has got three attributes namely: Staff_ID (As it’s the primary key of Staff), Training_date and Prof_level as per the statement in the assignment.
- Primary keys: The primary key for ‘Doctor’ is ‘Staff_ID’ as that will not be the same ever for any two tuples in the ‘Doctor’.
- Constraints: We have used the ‘Not Null’ in ‘Staff_ID’ to make sure we do not get any null values for it.
- Design Speciality: We have created ‘Dietician’ as the sub class from ‘Doctor’ super class using ISA relationship.

8. Speciality:

- Entity type: We have used a strong entity type for ‘Speciality’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Speciality’ has a Many-to-One relationship with ‘Doctor’. Here in this case, a doctor can have more than one speciality but not more than five.
- Attributes: ‘Speciality’ has only one attribute which is ‘Speciality’.
- Primary keys: Here we have created a composite primary key for ‘Speciality’ which is ‘Speciality’, and ‘Staff_ID’. ‘Staff_ID’ is created as foreign key from the ‘Staff’ entity.
- Constraints: We have created a trigger function which allows doctors to have at least one medical specialty but not more than five.

9. Interpreter:

- Entity type: We have used a strong entity type for ‘Interpreter’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Interpreter’ is one of the specialisations of the entity ‘Staff’ so, it is One-to-One in relationship with ‘Staff’.
- Attributes: ‘Interpreter’ has got two attributes namely: Staff_ID (As it’s the primary key of Staff), and Training_date as per the statement in the assignment.
- Primary keys: The primary key for ‘Interpreter’ is ‘Staff_ID’ as that will not be the same ever for any two tuples in the ‘Interpreter’.
- Constraints: We have used the ‘Not Null’ in ‘Staff_ID’ to make sure we do not get any null values for it.
- Design Speciality: We have created ‘Interpreter’ as the sub class from ‘Staff’ super class using ISA relationship.

10. Nurse:

- Entity type: We have used a strong entity type for ‘Nurse’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Nurse’ is one of the specialisations of the entity ‘Staff’ so, it is One-to-One in relationship with ‘Staff’.
- Attributes: ‘Nurse’ has got four attributes namely: Staff_ID (As it’s the primary key of Staff), WorkWithChildren, WWCC_IssueDate, WWCC_ExpiryDate as per the statement in the assignment.
- Primary keys: The primary key for ‘Nurse’ is ‘Staff_ID’ as that will not be the same ever for any two tuples in the ‘Nurse’.
- Constraints: We have used a ‘Check’ constraint that checks whether WorkWithChildren is False or True, also if it is True then whether (WWCC_ExpiryDate - WWCC_IssueDate) is less than 3 years/1095 days.
- Design Speciality: We have created ‘Nurse’ as the sub class from ‘Staff’ super class using ISA relationship.

11. Patient:

- Entity type: We have used a strong entity type for ‘Patient’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Patient’ has three One-to-One relationships with ‘Admission’ because each instance of ‘Patient’ can consist of exactly one instance of ‘Admission’.
- Attributes: ‘Patient’ has got nine attributes namely: Patient_ID, DoB, Full_name, Email, Address, EC_Phone, EC_Name, Insurance_No, Mobile as per the statement in the assignment.
- Primary keys: The primary key for ‘Patient’ is ‘Patient_ID’ as that will not be the same ever for any two tuples in the ‘Patient’.
- Constraints: We have used the ‘Not Null’ in Patient_id to make sure we do not get any null values for it.

12. Admission:

- Entity type: We have used a strong entity type for ‘Admission’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Admission’ has only one One-to-One relationship with ‘Patient’ because each instance of ‘Admission’ can consist of only one instance of ‘Patient’ as one patient is registered by one admission.
- Attributes: ‘Staff’ has got eight attributes namely: Admission_ID, Admission_time, Admission_date, Admission_type, Nurse_name, Doctor_name, Ref_practioner and Ref_no.
- Primary keys: The primary key for ‘Staff’ is ‘Admission_ID’ as that will not be the same ever for any two tuples in the ‘Nurse’.
- Constraints: We have used the ‘Not Null’ in ‘Admission_ID’ to make sure we do not get any null values for it. We have also used the ‘Check’ constraint that checks whether ‘Admission_type’ is Planned or Emergency. If it is Emergency then ‘Ref_practioner’ and ‘Ref_no’ can accept null values.

13. Billing:

- Entity type: We have used a strong entity type for ‘Billing’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Billing’ has a One-to-One relationship with ‘Admission’ because each instance of ‘Billing’ can consist of exactly one instance of ‘Admission’.
- Attributes: ‘Billing’ has got seven attributes : Billing_ID(As it’s the primary key of Billing), Payment_status,Remaining_Balance,Discharge_date,Total_cost,Service_Description and Insurance_Coverage as per the statement in the assignment.
- Primary keys: The primary key for ‘Billing’ is ‘Billing_ID’ as that will not be the same ever for any two tuples in the ‘Billing’.
- Constraints: We have used the ‘Not Null’ in ‘Billing_ID’ to make sure we do not get any null values for it.

14. Credit Card:

- Entity type: We have used a strong entity type for ‘Credit_card’ because it does not depend upon the existence of any other entity. Its existence is independent.
- Relationship type: ‘Credit_card’ has only one One-to-One relationship with ‘Billing’ because each instance of ‘Credit_card’ can consist of only one instance of ‘Billing’ as a bill can be paid with only one credit card.
- Attributes: ‘Credit_card’ has got four attributes namely: Card_No, DoB, Cardholder_name, CVV, Exp_date as per the statement in the assignment.
- Primary keys: The primary key for ‘Credit_card’ is ‘Card_No’ as that will not be the same ever for any two tuples in the ‘Credit_card’.
- Constraints: We have used the ‘Unique’ in billing_id to ensure a One-to-One relationship with Billing_ID. Also, We have used a ‘Check’ constraint that checks whether expiry date has passed the current date.