**Q6\_Part C**

Read the following article: https://vertabelo.com/blog/naming-conventions-in-database-modeling/

What naming conventions are recommended when creating a physical database. Give examples.

**Answer:** the combination of naming conventions and consistency is the best way to name the database.

Naming table:

* Be descriptive
* Use underscores to prefix a table name. for example, Teacher\_Math, NoTeacher\_Admin
* Avoid using DBMS-specific keywords as names for your tables like “Order”.
* Stick to one case whichever looks readable and easy on the eyes, either it be PascalCase or camelCase or all lowercase etc.

Naming Columns is corresponding to its use. Column names are like adjectives or verbs.

* Be unique. If two columns from different tables serving different purposes are in the same database then use some kind of prefixes that separate the two.
* Be abstract or cryptic. Use long descriptive names instead of short and unclear abbreviations.
* Not be very generic. For example, we can use Pens\_Code, instead of Code.

Naming Primary Keys

Primary keys serve as the unique identifier for your table, thus it is important to be careful while naming them. instead of naming a primary key representing an identity number as ID, we can use domain-specific names like Student\_ID

Naming Foreign Keys :

* Foreign keys are used as a bridge between two tables, and sometimes among more tables. Thus, it is good to name the foreign key with the same name consistently throughout the database to avoid any confusion.
* Like primary keys, we can add prefix or suffix to a foreign key name, like Subjectid\_fk

**2: Established business rules**

According to the concept ERD, the business rule is

1a. One department has one or more courses

1b. One course is offered by one department

2a. One course has one or more students

2b. One student takes one or more courses

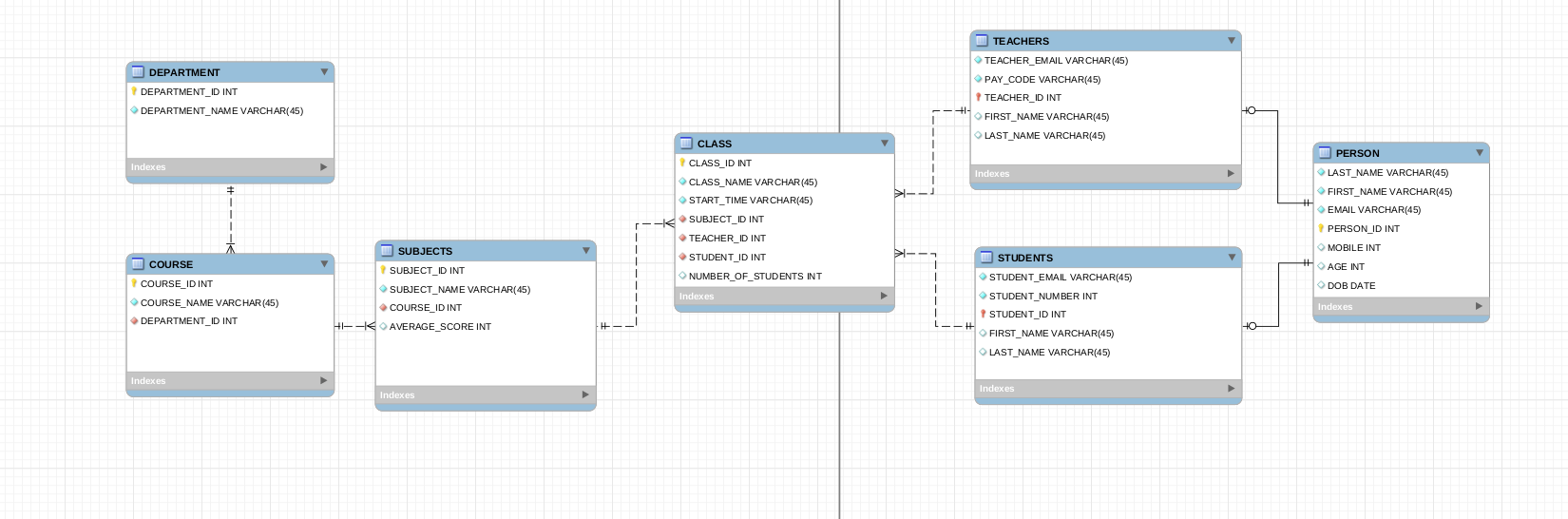
3a. One course has one or more teachers

3b. One teacher teaches one or more courses

4a. One teacher teaches one or more students

4b. One student has one or more teachers

**4 Developed a detailed ER diagram (identifying primary / foreign key relationships and attributes)**



All fields are not null.

All primary key is INT and auto\_ increment.

**5 Listed attributes for each table (data dictionary clearly indicating primary and foreign key relationships)**

*The above can be summarized for each table as shown below:*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table Name *–*  (tblTable \_ PERSONS) | | | | | | |
| Attribute | PK/FK | Data Type | Length | Acceptable Values | Default | Null Values Allowed |
| PERSON\_ID | PK | INT |  |  |  | N |
| FIRST\_NAME |  | VARCHAR | 45 |  |  | N |
| LAST\_NAME |  | VARCHAR | 45 |  |  | N |
| EMIAL |  | VARCHAR | 45 |  |  | N |
| MOBIEL |  | INT |  |  |  | N |
| AGE |  | INT |  |  |  | N |
| DOB |  | DATE |  |  |  | N |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table Name *–*  (tblTable \_ TEACHERS) | | | | | | |
| Attribute | PK/FK | Data Type | Length | Acceptable Values | Default | Null Values Allowed |
| TEACHER\_ID | PK/FK | INT |  |  |  | N |
| TEACHER\_EMAIL |  | VANCHAR | 45 |  |  | N |
| PAY\_CODE |  | VANCHAR | 45 |  |  | N |
| LAST\_NAME |  | VANCHAR | 45 |  |  | N |
| FIRST\_NAME |  | VANCHAR | 45 |  |  | N |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table Name *– tblTableName* (tblTable \_ COURSES) | | | | | | |
| Attribute | PK/FK | Data Type | Length | Acceptable Values | Default | Null Values Allowed |
| COURSE\_ID | PK | INT |  |  |  | N |
| COURSE\_NAME |  | VARCHAR | 45 |  |  | N |
| DEPARTMENT\_ID | FK | INT |  |  |  | N |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table Name *–*  (tblTable STUDENTS) | | | | | | |
| Attribute | PK/FK | Data Type | Length | Acceptable Values | Default | Null Values Allowed |
| STUDENT\_ID | PK/FK | INT |  |  |  | N |
| STUDENT\_EMAIL |  | VARCHAR | 45 |  |  | N |
| STUENDT\_NUMBER |  | INT |  |  |  | N |
| LAST\_NAME |  | VANCHAR | 45 |  |  | N |
| FIRST\_NAME |  | VANCHAR | 45 |  |  | N |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table Name *–*  (tblTable \_ SUBJECTS) | | | | | | |
| Attribute | PK/FK | Data Type | Length | Acceptable Values | Default | Null Values Allowed |
| SUBJECT ID | PK | INT |  |  |  | N |
| SUBJECT\_NAME |  | VARCHAR | 45 |  |  | N |
| AVERAGE\_SCORE |  | INT |  |  |  | N |
| COURSE\_ID | FK | INT |  |  |  | N |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table Name *– tblTableName* (tblTable \_ CLASS) | | | | | | |
| Attribute | PK/FK | Data Type | Length | Acceptable Values | Default | Null Values Allowed |
| CLASS\_ID | PK | INT |  |  |  | N |
| TEACHER\_ID | FK1 | INT |  |  |  | N |
| SUBJECT\_ID | FK2 | INT |  |  |  | N |
| STUDENT\_ID | FK3 | INT |  |  |  | N |
| START\_TIME |  | VARCHAR | 45 |  |  | N |
| CLASS\_NAME |  | VARCHAR | 45 |  |  | N |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table Name *– tblTableName* (tblTable \_ DEPARTMENT) | | | | | | |
| Attribute | PK/FK | Data Type | Length | Acceptable Values | Default | Null Values Allowed |
| DEPARTMENT\_ID | PK | INT |  |  |  | N |
| DEPARTMENT\_NAME |  | VARCHAR | 45 |  |  | N |

**Review your design and ensure that it satisfies all the business rules.**

After completing the detail ERD, I revised the business rule

there are 2 \* one to one relationship

1a:One person can be one teacher

1b:A teacher must be a person

2a:One person can be one student

2b:One student must be one person

5\* one to many relationship

1a: One course has one or more subjects

1b: One subject belongs to only one course

2a:One department has one or more courses

2b:One course is offered by one department

*Class is the joint table which has student\_id, subject\_id, teacher\_id, 3 FKs, together.*

3a: One subject has one or more classes

3b: One class only teaches one subject

4a:One teacher teaches one or more classes

4b:One class has only one teacher

5a:One student enrolls one or more classes

5b:One class has one student

3\* many to many relationship

1a:One subject has one or more students

1b:One student takes one or more subjects

2a:One subject has one or more teachers

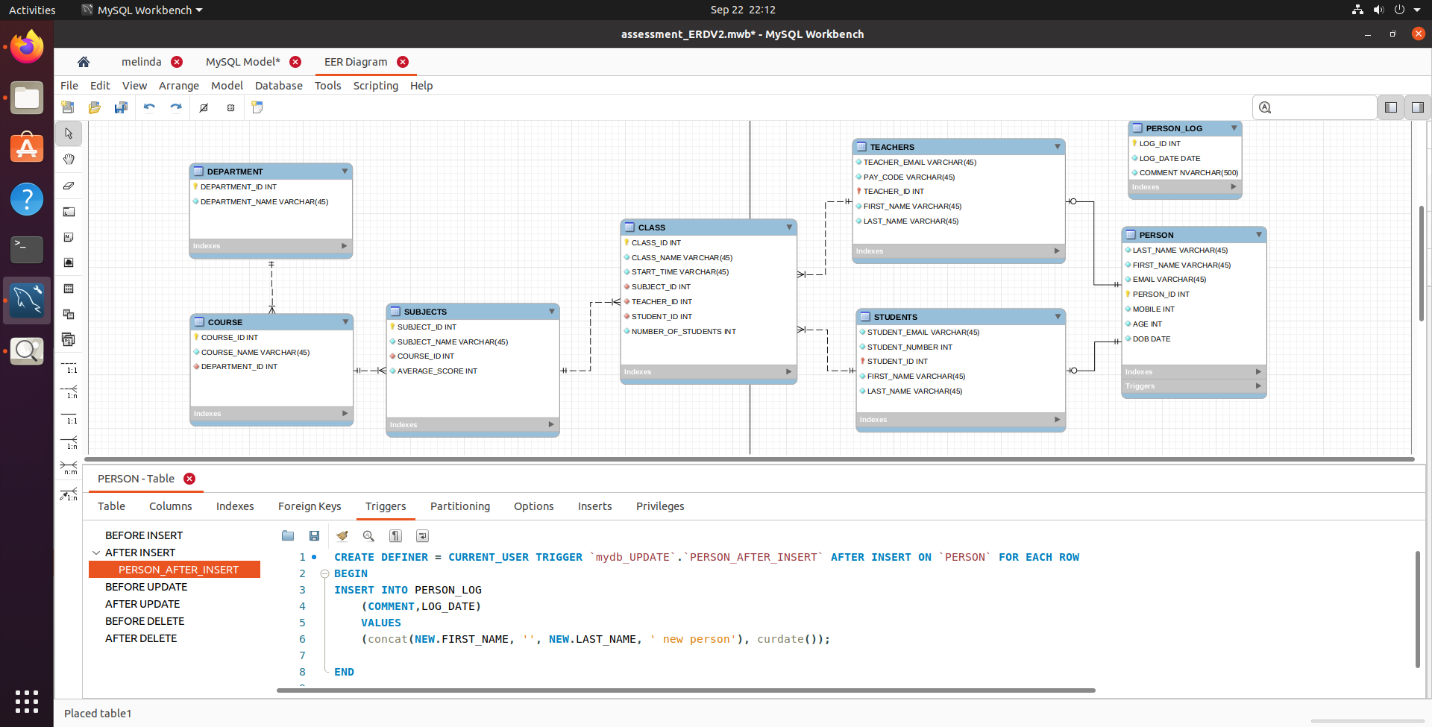
2b:One teacher teaches one or more subjects

3a:One teacher teaches one or many students.

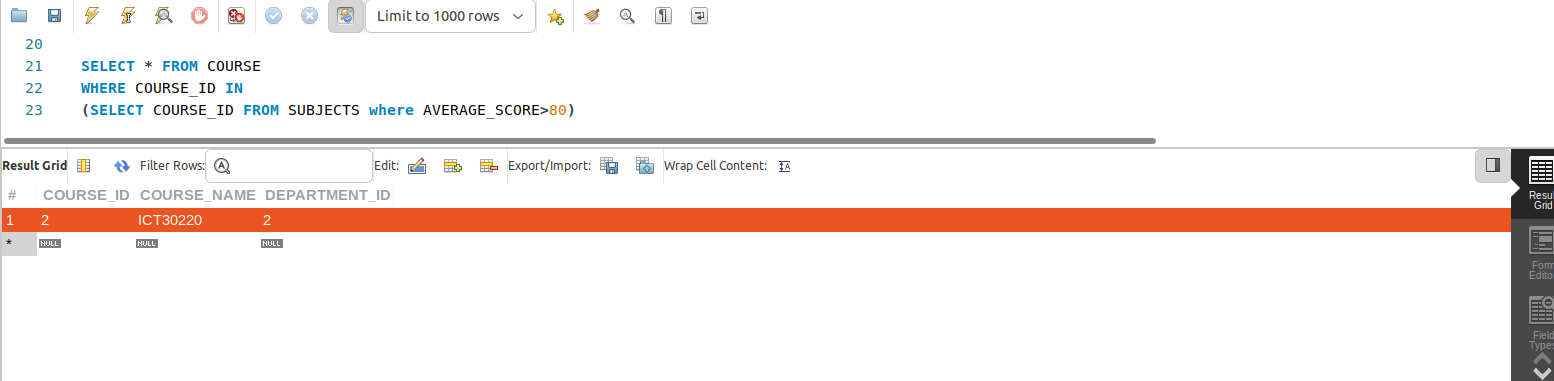
3b:One student has one or many teachers.

AND

After reviewing my design, I noticed I should have a person log table to automatically record when a new person added into the system. I added the table and trigger accordingly. See the screen shot please.



**12 Subquery**



SELECT \* FROM COURSE

WHERE COURSE\_ID IN

(SELECT COURSE\_ID FROM SUBJECTS where AVERAGE\_SCORE>80)