Database Design and SQL

Chapter 4: Normalization

FSDM/CPCM-2023S

Sagara Samarawickrama

Chapter Objectives

- Explain the term normalization
- Explain the different normal forms
- Use 1NF,2NF,and 3NF.

Normalization

In addition to entity relationship diagrams(ERD), normalization is a very valuable part of the database development process. Normalization is a series of used to evaluate and modify table structures to ensure that every non-key column in every table is directly dependent on the primary key

The results of normalization are reduced redundancies, fewer anomalies, and improved efficiencies. Normalization is another process toward a good database design. Normalization has two purposes

- Eliminate redundant data; that is, eliminate the storing of the same data in more than one table
- Ensure that data within a table are related

Eliminating redundant data is achieved by splitting tables with redundant data into two or more tables without redundancy.

Normalization

Normalization involves the process of applying rules called **normal forms** to table structures that produce a design that is free of data redundancy problems.

The steps for normalization are called normal forms , abbreviated as NF. Several normal forms exist, with the most common being first normal form **(1NF)** , second normal form**(2NF)**, and third normal form (**3NF)**. Each normal form addresses the potential for a particular type of redundancy and a table is said to be in one of the normal forms if it satisfies the rules required by that form.

Normalization

Normal form (NF)	Description
First normal form (1NF)	• No repeating groups; an entity (table) does not contain two or more columns that are closely related and thus provide similar data • No multivalued columns; each column can contain only one value • Primary key is identified.
Second normal form (2NF)	• 1NF. • No partial dependencies; all non-key columns are fully dependent on the entire primary key.
Third normal form (3NF)	• 2NF. • A non-key column cannot determine the value of another non-key column. Every non-key column must depend directly on the primary key.
Boyce-Codd normal form (BCNF) (sometimes referred to as 3.5NF)	• Every determinant in a table is a candidate key. If there is only one candidate key, 3NF and BCNF are the same.

Normalization

Representing Database Tables:

There are several ways to represent database tables during normalization process. The two methods we will be using in this course are, table form and relational schema.

Table Form :

- Capitalize entity names
- Bold and underline the primary key
- Italicize foreign keys

DEPARTMENT	
dept_id	dept_name
275	Sales
486	Manufacturing
694	Information Systems

EMPLOYEE			
emp_id	first_name	last_name	*dept_id*
111	Robert	Jackson	486
222	Betty	Rogers	486
333	Kumar	Patel	275

Normalization

Relational Schema:

A relational schema is a text-based method to represent database tables. In this method each table is its name , followed by a list of the table attributes in parenthesis.

- Capitalize the entity name
- Put attributes in parenthesis
- Bold and underline the primary key
- Italicize foreign keys

DEPARTMENT (**dept_id**, dept_name)

EMPLOYEE(**emp_id**, first_name, last_name, *dept_id*)

Normalization

Functional Dependency:

A functional dependency occurs when one attribute in a table uniquely determines another attribute. This can be written product_id -> prod_desc , which is the same as stating “product_desc is functionally dependent upon depart_id.”.The value of product_desc cannot be determined without knowing the value of product_id.

INVENTORY								
whse_id	**product_id**	prod_desc	bin	qty	whse_address	city	state	zip
111	167	Shovel	15	10	1511 Central Ave	Detroit	MI	48220
111	448	Hammer	88	26	1511 Central Ave	Detroit	MI	48220
222	167	Shovel	24	20	6803 Alder St	Dallas	TX	97335
222	302	Rake	21	18	6803 Alder St	Dallas	TX	97335

Task :

Identify the another functional dependency from the above table

Normalization

First Normal Form (1NF):

A table is in first normal form (1NF) when

- **No repeating groups**
- **No multivalued columns**
- **A primary key has been defined**
- **All columns in the table are dependent on primary key.**

Normalization

No Repeating Groups:

A repeating group means that an entity (tables) contains two or more columns that are closely related and thus provide similar data. For example , the following BOOK entity contains data on books and their authors.

Note : *This entity is not in 1NF because the column author_1,author_2 and author_3 are repeating similar data.*

BOOK					
book_id	book_title	author_1	author_2	author_3	price
101	My Database Book	Sally Jones	Peter Smith		65.00
102	Modern Programming	James Allen	Tim Peters	John Aims	84.00
103	Web Technology	Ken Albert			79.00

The normalized BOOK data can be brought into 1NF by consolidating the three author into one column as shown below.

Normalization

BOOK			
book_id	book_title	**author**	price
101	My Database Book	Sally Jones	65.00
101	My Database Book	Peter Smith	65.00
102	Modern Programming	James Allen	84.00
102	Modern Programming	Tim Peters	84.00
102	Modern Programming	John Aims	84.00
103	Web Technology	Ken Albert	79.00

Another method to resolve repeating groups is to create a new entity for the repeating group.

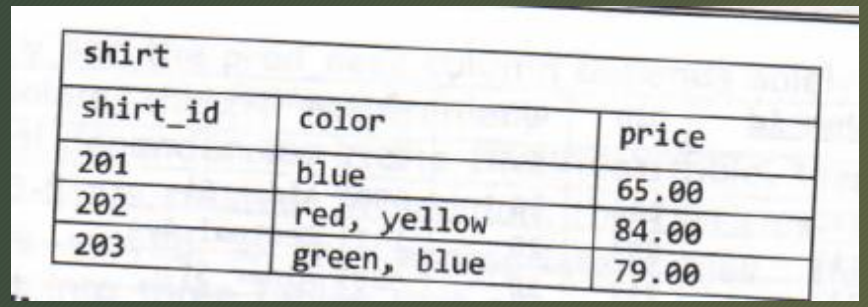
BOOK		
book_id	book_title	price
101	My Database Book	65.00
102	Modern Programming	84.00
103	Web Technology	79.00

AUTHOR		
author_id	author_name	*book_id*
11	Sally Jones	101
12	James Allen	102
13	Ken Albert	103
14	Peter Smith	101
15	Tim Peters	102
16	John Aims	102

Normalization

No Multivalued Columns:

A column cannot contain multiple values. Every column must contain a single value; that is, every intersection of a row and column contains only one value. In the SHIRT entity shown below the color column contains multiple values.



shirt		
shirt_id	color	price
201	blue	65.00
202	red, yellow	84.00
203	green, blue	79.00

The SHIRT entity is not in 1NF because the color column column contain multiple values. The shirt entity can be brought into 1NF by removing the color column and creating a new COLOR entity

Normalization

Task :

Convert this table to 1NF

INVENTORY								
product_id	prod_desc	whse_id	bin	qty	whse_address	city	state	zip
167	Shovel	111 222	150 244	19 26	1511 Central Ave. 6803 Alder St.	Detroit Dallas	MI TX	48220 97338
448	Hammer	111	883	20	1511 Central Ave.	Detroit	MI	48220
302	Rake	222	212	18	6803 Alder St.	Dallas	TX	97338

Hint : repeating groups, Primary Key

Normalization

Solution :

INVENTORY

whse_id	product_id	prod_desc	bin	qty	whse_address	city	state	zip
111	167	Shovel	150	19	1511 Central Ave.	Detroit	MI	48220
111	448	Hammer	883	20	1511 Central Ave.	Detroit	MI	48220
222	167	Shovel	244	26	6803 Alder St.	Dallas	FL	97338
222	302	Rake	212	18	6803 Alder St.	Dallas	FL	97338

INVENTORY (**whse_id**, **product_id**, prod_desc, qty, bin, whse_address, city, state, zip)

Normalization

Second Normal Form (2NF):

Second normal form applies only to tables that have concatenated primary key. A database table is in the second normal form (2NF) when

- **It is in first Normal Form (1NF)**
- **There are no partial dependencies. That is, each non-key attribute depends on the entire primary key.**

Normalization

Partial Dependence:

Partial dependency means that a non-key is dependent on part of but not the entire primary key. In this example , the prod_desc column is dependent on the product-id key but has no connection with the whsc_id key. Likewise , the whse_address column is dependent on the whsc_id key and has no connection with the product_id key. Thus, the prod_desc and whsc_address non-key columns do not depend on the entire composite primary key of product_id and whsc_id.

The primary key for the INVENTORY table (below) includes both the whse_id and product_id columns(neither alone is unique).Because the whse_address column depends solely on the whsc_id key and the prod_desc column depends solely on the product_id key, the table shown violates the 2NF environment.

whse_id	product_id	prod_desc	bin	qty	whse_address	city	state	zip
111	167	Shovel	150	19	1511 Central Ave.	Detroit	MI	48220
111	448	Hammer	883	20	1511 Central Ave.	Detroit	MI	48220
222	167	Shovel	244	26	6803 Alder St.	Dallas	FL	97338
222	302	Rake	212	18	6803 Alder St.	Dallas	FL	97338

INVENTORY (whse_id, product_id, prod_desc, qty, bin, whse_address, city, state, zip)

Normalization

To remove the partial dependencies in the INVENTORY table, the PRODUCT and WAREHOUSE tables are created.

PRODUCT

product_id	prod_desc
167	Shovel
302	Rake
448	Hammer

WAREHOUSE

whse_id	whse_address	city	state	zip
111	1511 Central Ave.	Detroit	MI	48220
222	6803 Alder St.	Dallas	TX	97338

INVENTORY

whse_id	*product_id*	bin	qty
111	167	159	19
111	448	883	20
222	167	244	26
222	302	212	18

PRODUCT (product_id, prod_desc)

WAREHOUSE (whse_id, whse_address, city, state, zip)

INVENTORY (*whse_id*, *product_id*, bin, qty)

Normalization

Referential Integrity:

According to the previous figure a separate warehouse table is defined so that a warehouse address can be stored non- redundantly. The whse_id is stored as a column in the WAREHOUSE and INVENTORY tables so the whsc_id value can be used in an INVENTORY row to reference , or look up , the appropriate WAREHOUSE row via the WAREHOUSE table whsc-id primary key. Thus the rows in the two tables are interrelated, based on matching values in the whsc_id columns in the two tables

The whsc_id column in the WAREHOUSE table serves as a primary key and can never be null. the whsc_id column in the INVENTORY table is referred to as a foreign key; it addresses “foreign” rows that are usually outside the same table. A foreign key value can be null .This means that its related row is unknown. A foreign key value also can match exactly a primary key value in a related row. But a foreign key value cannot have some column values present and not match the primary key value of an existing row in the related table

Normalization

Third Normal Form(3NF):

A database table is in third normal form (3NF) when;

- **It is in second normal form (2NF).**
- **The tables does not contain any non-key dependencies . A non key dependency also referred to as transitive dependency , occurs when a non-key attribute determines the value of another non-key attribute. To conform to 3NF , every attribute must depend only on the primary key.**

Lets consider the WAREHOUSE table shown below. This WAREHOUSE table is a result of the second normal form step, so it satisfies the first rule of 3NF

WAREHOUSE				
whse_id	whse_address	city	state	zip
111	1511 Central Ave.	Detroit	MI	48220
222	6803 Alder St.	Dallas	TX	97338

Normalization

The second rule of third normal for is that each non-key column depend only on the primary key and not another non-key column. Consider the city and the state columns .Do city and state depend on the primary key whsc_id? In this application, the whsc_id column is just an assigned number to identify each warehouse within the company . On the other hand , the zip column is used by the post office to identify city and state by Zip code. Thus , it can be said that the city and state columns are depend on the zip column. To satisfy the second rule of 3NF the warehouse table can be split into two tables as shown below.

WAREHOUSE

whse_id	whse_address	*zip*
111	1511 Central Ave.	48220
222	6803 Alder St.	97338

ZIP

zip	city	state
48220	Detroit	MI
97338	Dallas	TX

WAREHOUSE (**whse_id**, whse_address, *zip*)

ZIP (**zip**, city, state)

Normalization

Another example !

ORDER					
order_id	cust_id	product_id	quantity	price	total
601	123	132	11	25.86	284.46
602	789	546	54	35.77	1931.58
603	123	758	33	22.10	729.30
604	198	910	87	91.59	7968.33

is this table in 3NF ? If not ,Why? (hint: total_amt & primary key dependency)

ORDER				
order_id	cust_id	product_id	quantity	unit_price
601	123	132	11	25.86
602	789	546	54	35.77
603	123	758	33	22.10
604	198	910	87	91.59

Understanding database normalization technique is important to achieve a good database design. **If the database design does not confirm to at least the 3NF it may be difficult to achieve a successful database application.**

Normalization

Boyce- Codd Normal Form(BCNF): A table is in Boyce-Codd normal form(BCNF) if every determinant is a candidate key. A determinant is any column within the row that determines the value of another column. The BCNF was develop to handle situation in which a non-key columns determines the value of part of the primary key.

Blue water Hospital assigns assessment rooms by the day so visiting doctors can meet with patients. Mary in the assessment office schedules the doctor visits and assigns each doctor to specific assessment room . The schedule is posted on a board in the assessment entrance so doctors know which room they have been assigned for the day and patients know where their visit is scheduled. Mary wants this schedule to be normalized.

Normalization

APPOINTMENT				
patient id	appt date	app time	doctor	room id
38963	2020-05-16	10:45 AM	D142	E104
54321	2020-05-17	9:30 AM	D987	E101
83691	2020-05-17	9:30 AM	D142	E102
66301	2020-05-17	10:45 AM	D987	E101
54321	2020-05-17	12:15 PM	D639	E104
54321	2020-05-17	3:20 PM	D987	E101
14682	2020-05-18	9:30 AM	D987	E105
73811	2020-05-19	10:45 AM	D987	E103
54321	2020-05-20	9:30 AM	D987	E101

Let us consider the functional dependencies of data, which are shown below.

patient_id, appt_date, appt_time → doctor_id, room_id
doctor_id, appt_date, appt_time → patient_id, room_id
room_id, appt_date, appt_time → patient_id, doctor_id

Normalization

These three functional dependencies are all candidate keys for this table. For example , if we know the values of patient_id,apt_date, and apt_time, we then know the values of doctor_id and room_id . Thus , they meet the requirements of BCNF. Now , consider the following functional dependency:

Doctor_id, appt_date --→ room_id

This functional dependency is not a candidate key for the table because the concatenated key(doctor_id and appt_date) does not uniquely identify each row of the table. As a consequence , the APPOINTMENT table may result in update in update anomalies . For example , if the room number assigned to doctor D987 on May 17, 2011, needs to change , three rows have to be updated in the table.

To meet the requirements of BCNF, the APPOINTMENT table needs to be changed. The violating functional dependency needs to be removed by creating two new tables called APPOINTMENT and DOCTOR_ROOM as shown in the figure.

Normalization

APPOINTMENT			
patient_id	appt_date	app_time	doctor_id
38963	2020-05-16	10:45 AM	D142
54321	2020-05-17	9:30 AM	D987
83691	2020-05-17	9:30 AM	D142
66301	2020-05-17	10:45 AM	D987
54321	2020-05-17	12:15 PM	D639
54321	2020-05-17	3:20 PM	D987
14682	2020-005-18	9:30 AM	D987
73811	2020-05-19	10:45 AM	D987
54321	2020-05-20	9:30 AM	D987

DOCTOR_ROOM		
patient_id	appt_date	room_id
D142	2020-05-16	E104
D987	2020-05-17	E101
D142	2020-05-17	E102
D639	2020-05-17	E104
D987	2020-05-18	E105
D987	2020-05-19	E103
D987	2020-05-20	E101

APPOINTMENT (**patient_id**, **appt_date**, **appt_time**, doctor_id)
FK doctor_id, app_date → DOCTOR_ROOM

DOCTOR_ROOM (**doctor_id**, **appt_date**, room_id)

A table is in BCNF when it is in 3NF & Every Determinant in the table is a candidate key

Normalization

Practical Example :

STUDENT							
student_id	student_name	credits	advisor_id	advisor_name	course_code	course_desc	grade
12345	Jane Smith	12	654	Shirley Jones	CIS 101 CIS 102 CIS 110	Logic XHTML Visual Basic	A B A
98765	Thomas Last	9	745	Terry Evans	BUS 101 ENG 101 ENG 102	Business I English I English II	B C C
56789	Robert Sim	12	654	Shirley Jones	ACC 101 ACC 102 ENG 101 ENG 102	Accounting I Accounting II English I English II	B A A A

Convert to 1NF → Remove multivalued columns , identify the primary key

Normalization

STUDENT							
student_id	student_name	credits	advisor_id	advisor_name	course_code	course_desc	grade
12345	Jane Smith	12	654	Shirley	CIS 101	Logic	A
12345	Jane Smith	12	654	Shirley	CIS 102	XHTML	B
12345	Jane Smith	12	654	Shirley	CIS 110	Visual Basic	A
98765	Thomas Last	9	745	Terry Evans	BUS 101	Business I	B
98765	Thomas Last	9	745	Terry Evans	ENG 101	English I	C
98765	Thomas Last	9	745	Terry Evans	ENG 102	English II	C
56789	Robert Sim	12	654	Shirley	ACC 101	Accounting I	B
56789	Robert Sim	12	654	Shirley	ACC 102	Accounting II	A
56789	Robert Sim	12	654	Shirley	ENG 101	English I	A
56789	Robert Sim	12	654	Shirley	ENG 102	English II	A

Now it is in 1NF because ; no repeating groups , no multivalued columns , primary key attributes defined

Now → Second Normal Form (2NF) … eliminate partial dependencies

Normalization

Eliminate partial dependencies.

Partial dependency exists when a column is determined by only part of the primary key. This occurs in a table that contains a concatenated primary key. To resolve partial dependency each component of the primary key identified in 1NF is used to create new table. As a result ,the original table is divided into three tables shown in the dependency diagram below.

Student_id -> student_name,credits,advisor_id,advisor_name

Course_code-> course_desc

STUDENT				
student_id	student_name	credits	advisor_id	advisor_name
12345	Jane Smith	12	654	Shirley Jones
98765	Thomas Last	9	745	Terry Evans
56789	Robert Sim	12	654	Shirley Jones

Normalization

Eliminate partial dependencies.

COURSE

course_code	course_desc
CIS 101	Logic
CIS 102	XHTML
CIS 110	Visual Basic
BUS 101	Business I
ENG 101	English I
ENG 102	English II
ACC 101	Accounting I
ACC 102	Accounting II

STUDENT_COURSE

student_id	course_code	grade
12345	CIS 101	A
12345	CIS 102	B
12345	CIS 110	A
98765	BUS 101	B
98765	ENG 101	C
98765	ENG 102	C
56789	ACC 101	B
56789	ACC 102	A
56789	ENG 101	A
56789	ENG 102	A

STUDENT (**student_id**, student_name, credits, advisor_id, advisor_name)
COURSE (**course_code**, course_desc)
STUDENT_COURSE (**student_id**, **course_code**, grade)
FK student_id → STUDENT
FK course_code → COURSE

Normalization

Third Normal Form(3NF)

This table is not in 3NF when a non-key column determine the value of another non-key column(s). As mentioned , a determinant is any column within the row that determines the value of another column. In the STUDENTS table, advisor_id is a determinant that determines the value of advisor name. As a result , a new table is created with the determinant as the primary key. In addition , the determinant attribute advisor_id remains in the original table as a foreign key

STUDENT			
student_id	student_name	credits	*advisor_id*
12345	Jane Smith	12	654
98765	Thomas Last	9	745
56789	Robert Sim	12	654

ADVISOR	
advisor_id	advisor_name
654	Shirley Jones
745	Terry Evans

Relational schema of tables in 3NF:

STUDENT (**student_id**, student_name, credits, *advisor_id*)
FK advisor_id → ADVISOR

ADVISOR (**advisor_id**, advisor_name)

Normalization

STUDENT (student_id, student_name, credits,advisor_id)

FK advisor_id -> ADVISOR

ADVISOR (advisor_id, advisor_name)

COURSE(course_id,course_desc)

STUDENT_COURSE (student_id,course_id,grade)

FK student_id -> STUDENT

FK course_id -> COURSE

The tables are now in 3NF because

- They are in 2NF
- They do not contain non-key columns that determine the value of other non-key columns

Normalization

STUDENT

student_id	student_name	credits	*advisor_id*
12345	Jane Smith	12	654
98765	Thomas Last	9	745
56789	Robert Sim	12	654

ADVISOR

advisor_id	advisor_name
654	Shirley Jones
745	Terry Evans

COURSE

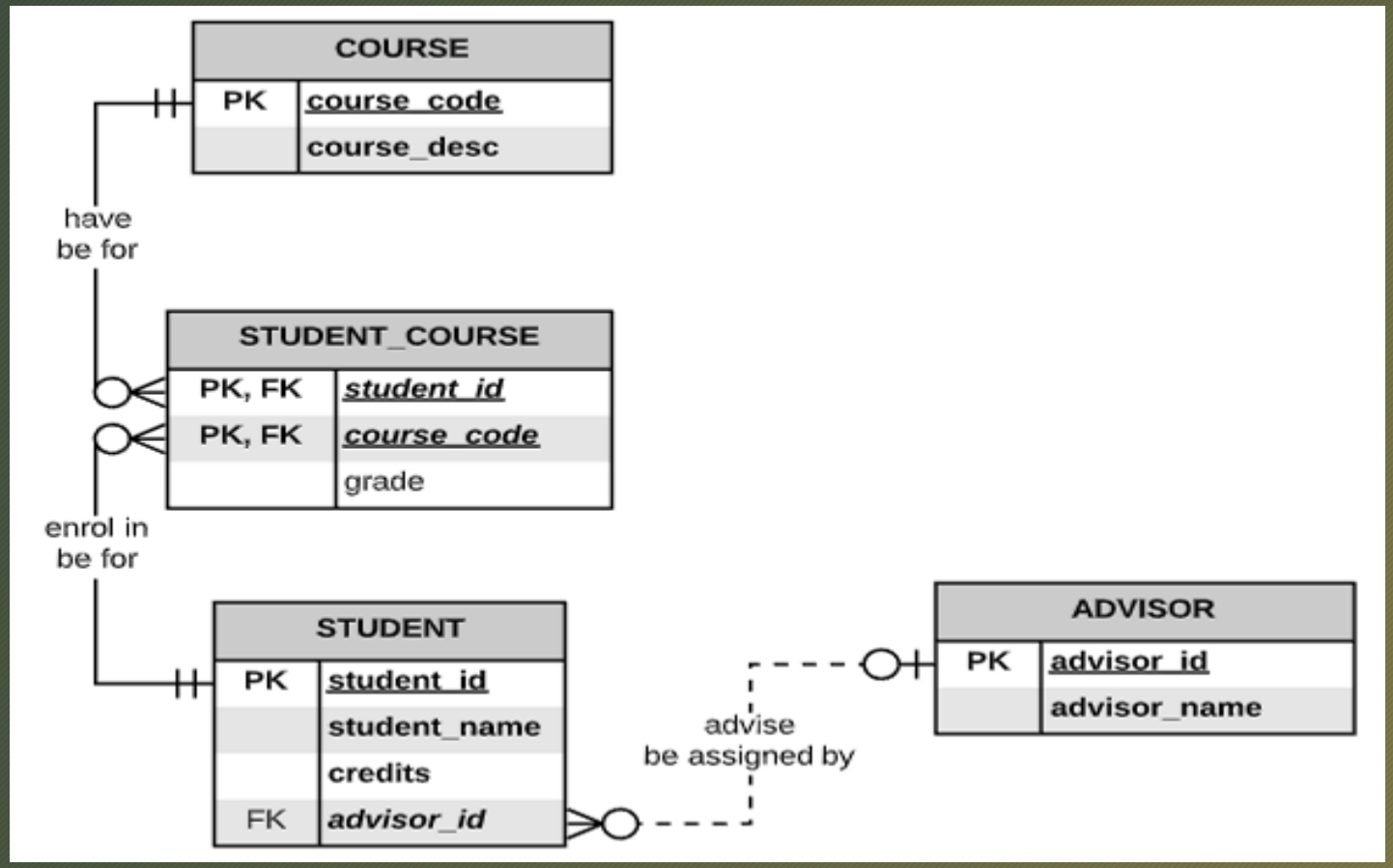
course_code	course_desc
CIS 101	Logic
CIS 102	XHTML
CIS 110	Visual Basic
BUS 101	Business I
ENG 101	English I
ENG 102	English II
ACC 101	Accounting I
ACC 102	Accounting II

STUDENT_COURSE

student_id	**course_code**	grade
12345	CIS 101	A
12345	CIS 102	B
12345	CIS 110	A
98765	BUS 101	B
98765	ENG 101	C
98765	ENG 102	C
56789	ACC 101	B
56789	ACC 102	A
56789	ENG 101	A
56789	ENG 102	A

STUDENT (**student_id**, student_name, credits, *advisor_id*)
  FK advisor_id → ADVISOR
ADVISOR (**advisor_id**, advisor_name)
COURSE (**course_code**, course_desc)
STUDENT_COURSE (**student_id**, **course_code**, grade)
  FK student_id → STUDENT
  FK course_code → COURSE

Normalization



Conclusion