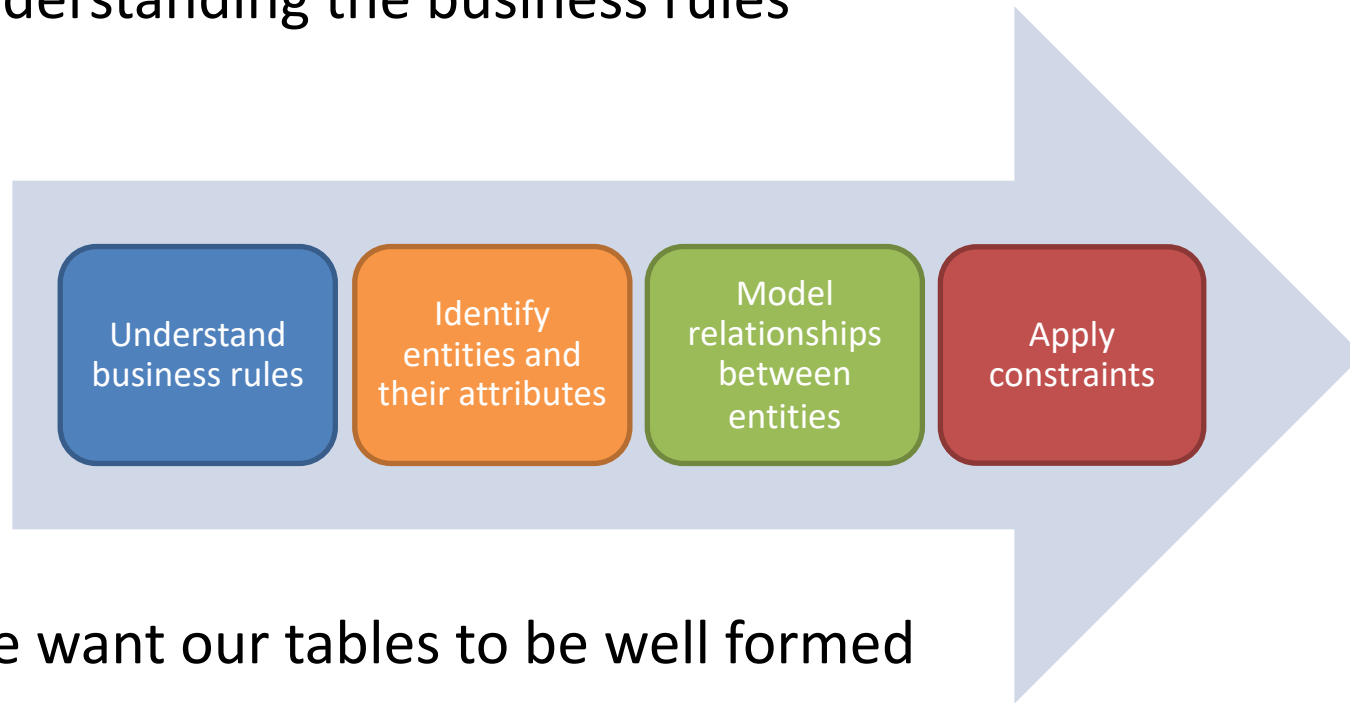


CS 61: Database Systems

Normalization

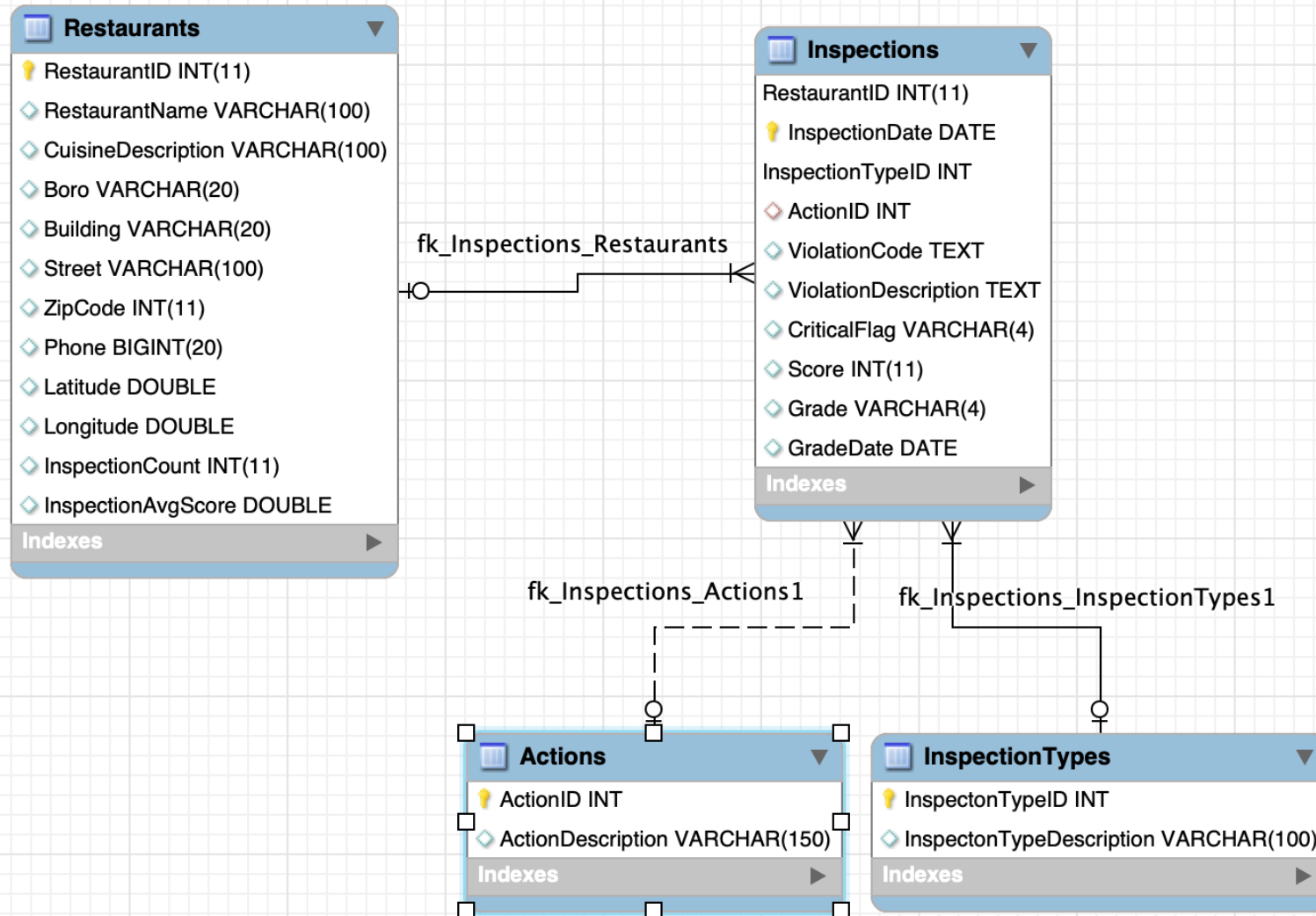
Objective: create well-formed relations

- Tables are the building blocks of a relational database
- Previously we created tables for entities identified after understanding the business rules



- We want our tables to be well formed
- Question, how do we know if our tables are well formed?
- It turns out a few relatively simple rules can help us

Version 1: Restaurants, Inspections and FK constraints on Action and Inspection Type



Normalization is the process by which we confirm our tables are well formed

Normalization is the process of evaluating and correcting poor table structure by following a few rules:

- Each table represents a single entity
- Each row/column intersection contains only one value and not a group of values
- No data item will be unnecessarily stored in more than one table
- All nonprime attributes (attributes not part of the key) in a table are dependent on the primary key
- Each table has no insertion, update, or deletion anomalies



The screenshot shows a database table named 'Restaurants'. The table has the following columns and data types:

Column Name	Data Type
RestaurantID	INT(11)
RestaurantName	VARCHAR(100)
CuisineDescription	VARCHAR(100)
Boro	VARCHAR(20)
Building	VARCHAR(20)
Street	VARCHAR(100)
ZipCode	INT(11)
Phone	BIGINT(20)
Latitude	DOUBLE
Longitude	DOUBLE
InspectionCount	INT(11)
InspectionAvgScore	DOUBLE

Below the columns, there is a section for 'Indexes' with a right-pointing arrow.

**Eliminate data anomalies
by removing unnecessary
or unwanted data
redundancies**

We will examine one table at a time, moving from First to Third Normal Form

First (1NF), Second (2NF) and Third (3NF) normal form characteristics



First Normal Form (1NF)

- Data in table format
- No repeating groups
- PK and all dependencies identified

Second Normal Form (2NF)

- 1NF plus
- No partial dependencies

Third Normal Form (3NF)

- 2NF plus
- No transitive dependencies

- **Work one relation at a time**
- **Progressively break relation into set of smaller relations as needed moving from 1NF to 3NF**

Higher forms mainly of academic interest only

Agenda

- 
1. Data anomalies
 2. Normalization

Database anomalies

Soccer player database

PlayerID	Name	Team	TeamPhone	Position1	Position2	Position3
1	Pessi	Argentina	54-11-1000-1000	Striker	Forward	
2	Ricardo	Portugal	351-2-7777-7777	Right Midfield	Defending Midfielder	
3	Neumann	Brazil	55-21-4040-2020	Forward	Left Fullback	Right Fullback
4	Baily	Wales	44-29-1876-1876	Defending Midfielder	Striker	
5	Marioso	Argentina	54-11-1000-1000	Sweeper	Defending Midfielder	Striker
6	Pare	Brazil	55-21-4040-2020	Goalkeeper		

Business rules

- Each player uniquely identified by PlayerID (it is a Primary Key here)
- Each player plays for one team and can play one or more position
- Each team has one phone number

Insert anomaly: can not add data due to absence of other data

Soccer player database

PlayerID	Name	Team	TeamPhone	Position1	Position2	Position3
1	Pessi	Argentina	54-11-1000-1000	Striker	Forward	
2	Ricardo	Portugal	351-2-7777-7777	Right Midfield	Defending Midfielder	
3	Neumann	Brazil	55-21-4040-2020	Forward	Left Fullback	Right Fullback
4	Baily	Wales	44-29-1876-1876	Defending Midfielder	Striker	
5	Marioso	Argentina	54-11-1000-1000	Sweeper	Defending Midfielder	Striker
6	Pare	Brazil	55-21-4040-2020	Goalkeeper		
∅	∅	Iceland	54-12-5432-2345	∅	∅	∅

Insert anomaly:

- Can't add team (say Iceland) without adding a player for that team because PlayerID is Primary Key
- Also no consistency in position names
 - What if some teams call a Sweeper a Center Back
 - How would we know they are the same?
- What if a player can play more than three positions?

Update anomaly: must update multiple tuples for one change

Soccer player database

PlayerID	Name	Team	TeamPhone	Position1	Position2	Position3
1	Pessi	Argentina	54-11-1000-1000	Striker	Forward	
2	Ricardo	Portugal	351-2-7777-7777	Right Midfield	Defending Midfielder	
3	Neumann	Brazil	55-21-4040-2020	Forward	Left Fullback	Right Fullback
4	Baily	Wales	44-29-1876-1876	Defending Midfielder	Striker	
5	Marioso	Argentina	54-11-1000-1000	Sweeper	Defending Midfielder	Striker
6	Pare	Brazil	55-21-4040-2020	Goalkeeper		
∅	∅	Iceland	54-12-5432-2345	∅	∅	∅

Update anomaly:

- If team moves, must update TeamPhone for all players on that team
- Could lead to inconsistency if some team players are updated, but not all

Delete anomaly: unintended loss of data

Soccer player database

PlayerID	Name	Team	TeamPhone	Position1	Position2	Position3
1	Pessi	Argentina	54-11-1000-1000	Striker	Forward	
2	Ricardo	Portugal	351-2-7777-7777	Right Midfield	Defending Midfielder	
3	Neumann	Brazil	55-21-4040-2020	Forward	Left Fullback	Right Fullback
4	Baily	Wales	44-29-1876-1876	Defending Midfielder	Striker	
5	Marioso	Argentina	54-11-1000-1000	Sweeper	Defending Midfielder	Striker
6	Pare	Brazil	55-21-4040-2020	Goalkeeper		
Ø	Ø	Iceland	54-12-5432-2345	Ø	Ø	Ø

Delete anomaly:

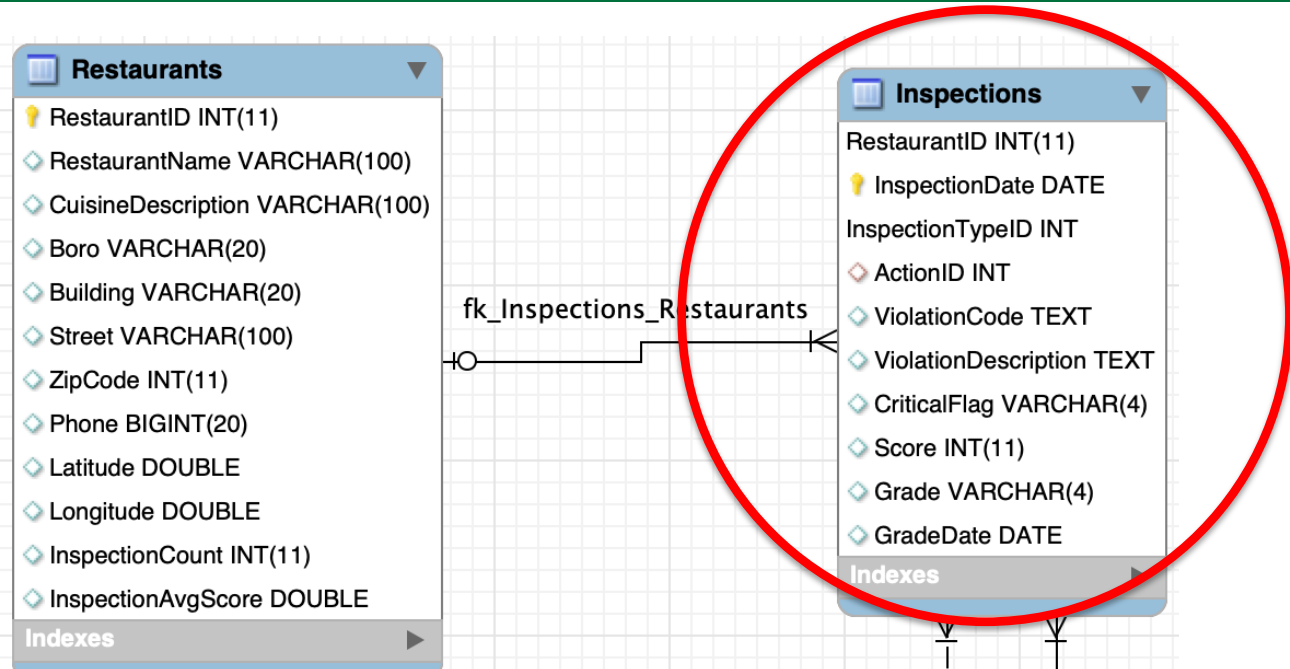
- If Ricardo retires, must remove from database
- If so, loose Portugal team data as well!

How does this apply to our inspection database?

Version 1: composite Primary Key can identify all rows in the Inspections table

Version 1

While a restaurant may receive more than one inspection on a given day (e.g. cycle and trans fat initial inspection), it will not receive more than one of the same type

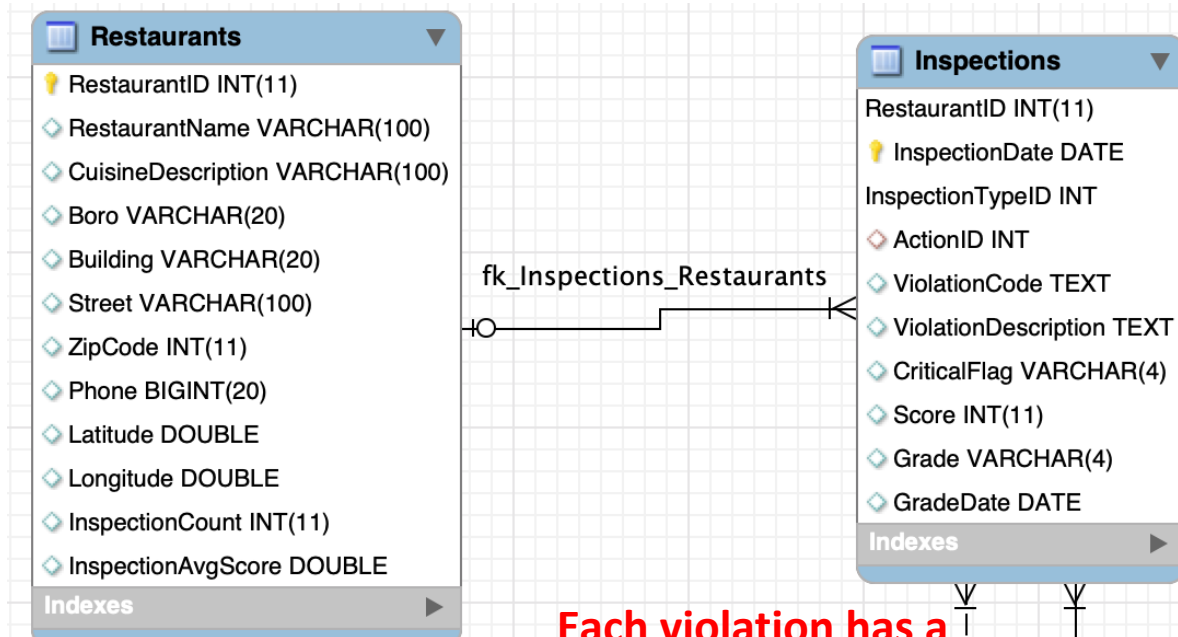


Inspection table: compound Primary Key uniquely identifies row: RestaurantID, InspectionDate, InspectionType

Restaurant...	InspectionD...	InspectionType	ActionDescription	ViolationCode	ViolationDescription	CriticalFlag	Score	Grade	GradeDate
30075445	2019-11-04	Cycle Inspection / Re-inspection	Violations were cited...	04N	Filth flies or food/refu...	Y	7	A	2019-11-04
30075445	2019-10-21	Cycle Inspection / Initial Inspection	Violations were cited...	08A, 06D, 04L, 10B	Facility not vermin pr...	Y	17	NULL	NULL
30075445	2019-06-11	Cycle Inspection / Re-inspection	Violations were cited...	08C, 10B, 10F	Pesticide use not in a...	N	6	A	2019-06-11
30075445	2019-05-16	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 04L, 08A	Non-food contact surf...	Y	14	NULL	NULL
30075445	2018-05-11	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 08C	Non-food contact surf...	N	5	A	2018-05-11
30075445	2017-05-18	Cycle Inspection / Initial Inspection	Violations were cited...	06D, 10F	Food contact surface...	Y	7	A	2017-05-18
30112340	2019-03-04	Cycle Inspection / Initial Inspection	Violations were cited...	02G, 10F, 10B	Cold food item held a...	Y	13	A	2019-03-04

Violation codes and descriptions can be rolled up into one field for each

Version 1



Is this a good design?

Let's take a closer look

CriticalFlag set to 'Y' if any violation is critical, otherwise 'N'

Each inspection can result in multiple violations

Each violation has a description (concatenated here)

Restaurant...	InspectionD...	InspectionType	ActionDescription	ViolationCode	ViolationDescription	CriticalFlag	Score	Grade	GradeDate
30075445	2019-11-04	Cycle Inspection / Re-inspection	Violations were cited...	04N	Filth flies or food/refu...	Y	7	A	2019-11-04
30075445	2019-10-21	Cycle Inspection / Initial Inspection	Violations were cited...	08A, 06D, 04L, 10B	Facility not vermin pr...	Y	17	NULL	NULL
30075445	2019-06-11	Cycle Inspection / Re-inspection	Violations were cited...	08C, 10B, 10F	Pesticide use not in a...	N	6	A	2019-06-11
30075445	2019-05-16	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 04L, 08A	Non-food contact surf...	Y	14	NULL	NULL
30075445	2018-05-11	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 08C	Non-food contact surf...	N	5	A	2018-05-11
30075445	2017-05-18	Cycle Inspection / Initial Inspection	Violations were cited...	06D, 10F	Food contact surface...	Y	7	A	2017-05-18
30112340	2019-03-04	Cycle Inspection / Initial Inspection	Violations were cited...	02G, 10F, 10B	Cold food item held a...	Y	13	A	2019-03-04

There are insert anomalies

Version 1

If a new type of violation code were created, it will not exist in the database until a restaurant gets this type of violation

Restaurants	
RestaurantID	INT(11)
RestaurantName	VARCHAR(100)
CuisineDescription	VARCHAR(100)
Boro	VARCHAR(20)
Building	VARCHAR(20)
Street	VARCHAR(100)
ZipCode	INT(11)
Phone	BIGINT(20)
Latitude	DOUBLE
Longitude	DOUBLE
InspectionCount	INT(11)
InspectionAvgScore	DOUBLE
Indexes	

fk_Inspections_Restaurants

Inspections	
InspectionID	INT
RestaurantID	INT(11)
InspectionDate	DATE
InspectionTypeID	INT
ActionID	INT
<u>ViolationCode</u>	TEXT
ViolationDescription	TEXT
CriticalFlag	VARCHAR(4)
Score	INT(11)
Grade	VARCHAR(4)
GradeDate	DATE
Indexes	

Also no consistency in values

Could enter a violation code that does not exist or a violation description that does not match the violation code

Restaurant...	InspectionD...	InspectionType	ActionDescription	ViolationCode	ViolationDescription	CriticalFlag	Score	Grade	GradeDate
30075445	2019-11-04	Cycle Inspection / Re-inspection	Violations were cited...	04N	Filth flies or food/refu...	Y	7	A	2019-11-04
30075445	2019-10-21	Cycle Inspection / Initial Inspection	Violations were cited...	08A, 06D, 04L, 10B	Facility not vermin pr...	Y	17	NULL	NULL
30075445	2019-06-11	Cycle Inspection / Re-inspection	Violations were cited...	08C, 10B, 10F	Pesticide use not in a...	N	6	A	2019-06-11
30075445	2019-05-16	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 04L, 08A	Non-food contact surf...	Y	14	NULL	NULL
30075445	2018-05-11	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 08C	Non-food contact surf...	N	5	A	2018-05-11
30075445	2017-05-18	Cycle Inspection / Initial Inspection	Violations were cited...	06D, 10F	Food contact surface...	Y	7	A	2017-05-18
30112340	2019-03-04	Cycle Inspection / Initial Inspection	Violations were cited...	02G, 10F, 10B	Cold food item held a...	Y	13	A	2019-03-04

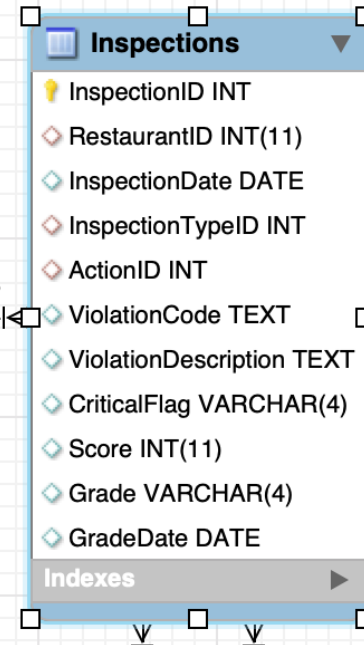
There are update anomalies

Version 1

If we change the description for a violation code, we must update the description in all rows with that description



fk_Inspections_Restaurants



Hard to find violation codes and descriptions as there are multiple entries in these columns

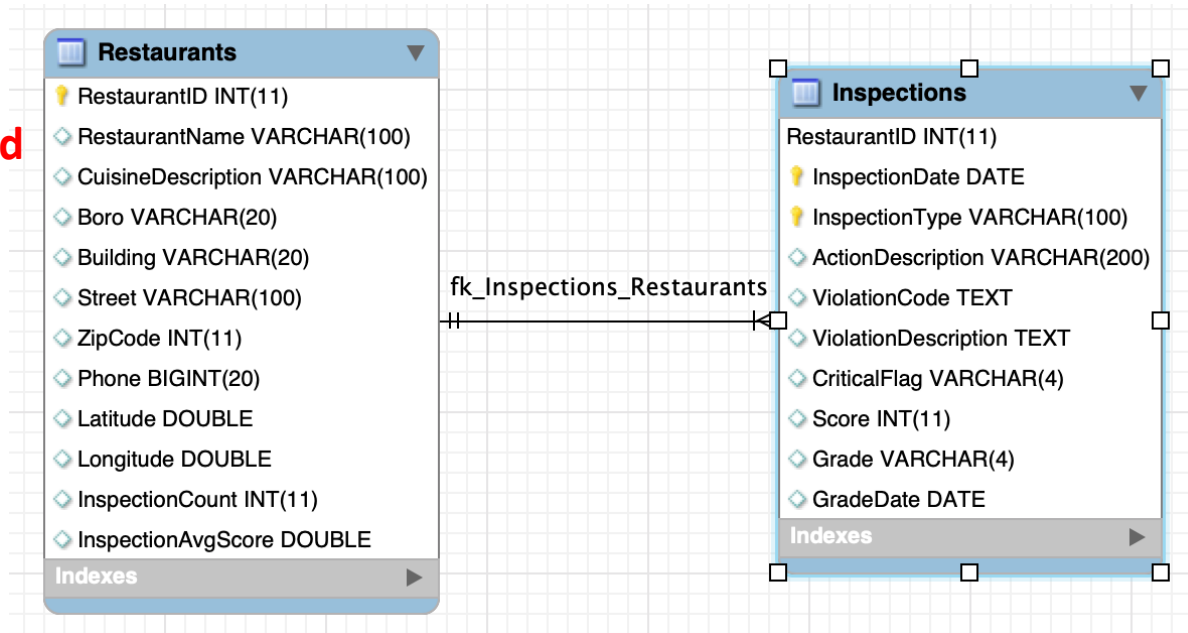
CriticalFlag also depends on all values in the ViolationCode column

Restaurant...	InspectionD...	InspectionType	ActionDescription	ViolationCode	ViolationDescription	CriticalFlag	Score	Grade	GradeDate
30075445	2019-11-04	Cycle Inspection / Re-inspection	Violations were cited...	04N	Filth flies or food/refu...	Y	7	A	2019-11-04
30075445	2019-10-21	Cycle Inspection / Initial Inspection	Violations were cited...	08A, 06D, 04L, 10B	Facility not vermin pr...	Y	17	NULL	NULL
30075445	2019-06-11	Cycle Inspection / Re-inspection	Violations were cited...	08C, 10B, 10F	Pesticide use not in a...	N	6	A	2019-06-11
30075445	2019-05-16	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 04L, 08A	Non-food contact surf...	Y	14	NULL	NULL
30075445	2018-05-11	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 08C	Non-food contact surf...	N	5	A	2018-05-11
30075445	2017-05-18	Cycle Inspection / Initial Inspection	Violations were cited...	06D, 10F	Food contact surface...	Y	7	A	2017-05-18
30112340	2019-03-04	Cycle Inspection / Initial Inspection	Violations were cited...	02G, 10F, 10B	Cold food item held a...	Y	13	A	2019-03-04

There are delete anomalies

Version 1

If only one inspection found a particular violation code, and we delete that inspection, we would lose the violation code



How do we fix this situation?

Normalization!

Restaurant...	InspectionD...	InspectionType	ActionDescription	ViolationCode	ViolationDescription	CriticalFlag	Score	Grade	GradeDate
30075445	2019-11-04	Cycle Inspection / Re-inspection	Violations were cited...	04N	Filth flies or food/refu...	Y	7	A	2019-11-04
30075445	2019-10-21	Cycle Inspection / Initial Inspection	Violations were cited...	08A, 06D, 04L, 10B	Facility not vermin pr...	Y	17	NULL	NULL
30075445	2019-06-11	Cycle Inspection / Re-inspection	Violations were cited...	08C, 10B, 10F	Pesticide use not in a...	N	6	A	2019-06-11
30075445	2019-05-16	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 04L, 08A	Non-food contact surf...	Y	14	NULL	NULL
30075445	2018-05-11	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 08C	Non-food contact surf...	N	5	A	2018-05-11
30075445	2017-05-18	Cycle Inspection / Initial Inspection	Violations were cited...	06D, 10F	Food contact surface...	Y	7	A	2017-05-18
30112340	2019-03-04	Cycle Inspection / Initial Inspection	Violations were cited...	02G, 10F, 10B	Cold food item held a...	Y	13	A	2019-03-04

Agenda

1. Data anomalies

 2. Normalization

Normalization is about correcting table structure to minimize data redundancy

Normalization

- Works in a series of stages called normal forms
- First normal form (1NF) through third normal form (3NF) or higher
- High forms tend to split relations into multiple relations, each with fewer attributes
- Generally the higher the form, the more joins are required to produce data
 - More resources required by the database to respond to requests
 - Slower performance
- Occasionally we will denormalize tables
 - Denormalization may result in redundant/dependent data
 - Particularly common in reporting/analysis databases
 - Deciding when to denormalize is part of the “art” of good database design

Key review

Key type	Definition
Superkey	An attribute or combination of attributes that uniquely identifies each row in a table
Candidate key	A minimal (irreducible) superkey; a superkey that does not contain a subset of attributes that is itself a superkey
Primary key	A candidate key selected to uniquely identify all other attribute values in any given row; cannot contain null entries
Foreign key	An attribute or combination of attributes in one table whose values must either match the primary key in another table or be null
Composite key	A key comprised of multiple attributes (sometimes called a compound key)

Functional dependence is a generalized notion of keys

Functional dependence (FD)

- One or more attributes determine the the value of one or more other attributes in a relation
- This role of a key — to determine the value of other attributes
- Written $A \rightarrow B$
 - A is called the **determinant**
 - B is called the **dependent** (value identified by another variable)
 - Here A is the (possibly composite) key and B is a collection of attributes that can be looked up given key A

Can look up B, if given A

Functional dependence can be full, partial or transitive

Full functional dependence

- An attribute is functionally dependent on a composite key but not any subset of the key (e.g., all attributes in key are required)

Ex: RestaurantID, InspectionDate, InspectionType $\rightarrow$ Score

All three attributes are required to uniquely identify the score

Partial dependence

- An attribute is dependent on only part of the key
Ex: RestaurantID, InspectionDate, InspectionType $\rightarrow$ InspectionDescription
- Only depends on InspectionType — straight forward, easy to identify

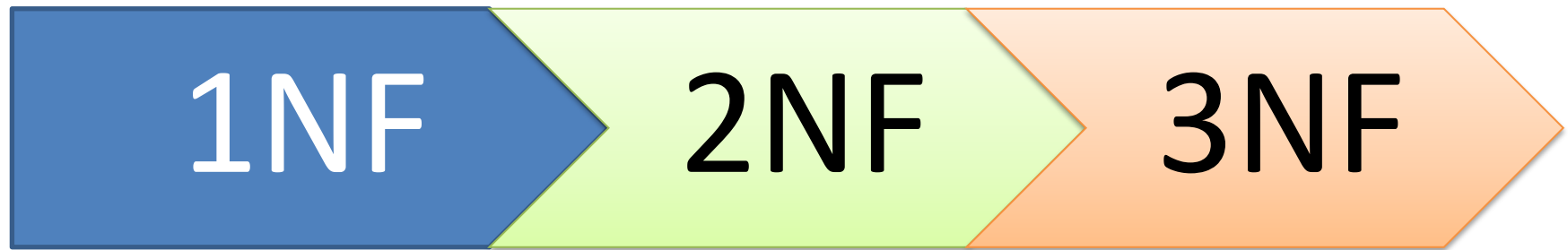
Transitive dependence

- If $A \rightarrow B$ and $B \rightarrow C$, then $A \rightarrow C$
- An attribute is dependent on another attribute that is not part of the key
- More difficult to identify among a set of data
- Occurs when functional dependence exists among nonprime attributes

Ex: ViolationCode $\rightarrow$ ViolationDescription, CriticalFlag

We will examine one table at a time, moving from First to Third Normal Form

First (1NF), Second (2NF) and Third (3NF) normal form characteristics



First Normal Form (1NF)

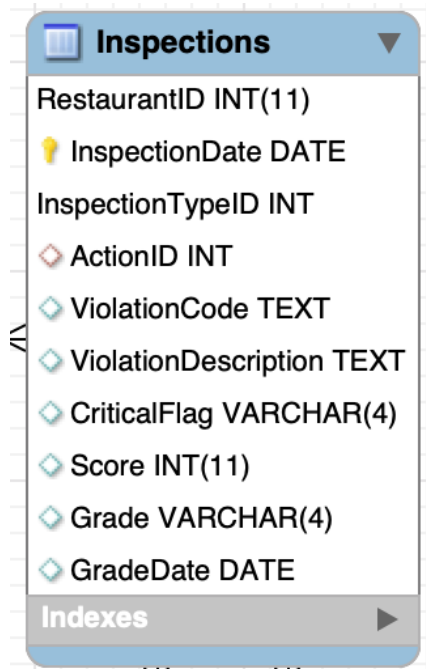
- Data in table format
- No repeating groups
- PK and all dependencies identified

Start with 1NF: table form, no repeating groups, PK and dependencies identified

Inspections table

Step 1: Put in table form

- Already done



The screenshot shows a database schema tool interface. At the top, there is a tab labeled 'Inspections' with a dropdown arrow. Below the tab, the table structure is listed with data types and constraints:

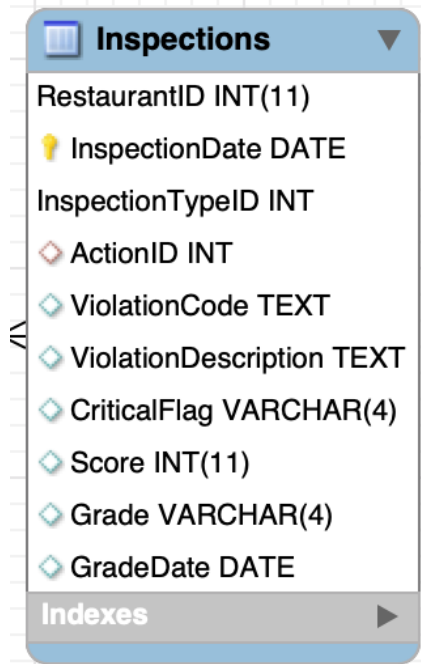
- RestaurantID INT(11)
- InspectionDate DATE (marked with a yellow lightning bolt icon, indicating a primary key)
- InspectionTypeID INT
- ActionID INT (marked with a red diamond icon, indicating a foreign key)
- ViolationCode TEXT (marked with a blue diamond icon, indicating a foreign key)
- ViolationDescription TEXT (marked with a blue diamond icon, indicating a foreign key)
- CriticalFlag VARCHAR(4) (marked with a blue diamond icon, indicating a foreign key)
- Score INT(11)
- Grade VARCHAR(4) (marked with a blue diamond icon, indicating a foreign key)
- GradeDate DATE (marked with a blue diamond icon, indicating a foreign key)

At the bottom of the list, there is a section labeled 'Indexes' with a right-pointing arrow.

Restaurant...	InspectionD...	InspectionType	ActionDescription	ViolationCode	ViolationDescription	CriticalFlag	Score	Grade	GradeDate
30075445	2019-11-04	Cycle Inspection / Re-inspection	Violations were cited...	04N	Filth flies or food/refu...	Y	7	A	2019-11-04
30075445	2019-10-21	Cycle Inspection / Initial Inspection	Violations were cited...	08A, 06D, 04L, 10B	Facility not vermin pr...	Y	17	NULL	NULL
30075445	2019-06-11	Cycle Inspection / Re-inspection	Violations were cited...	08C, 10B, 10F	Pesticide use not in a...	N	6	A	2019-06-11
30075445	2019-05-16	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 04L, 08A	Non-food contact surf...	Y	14	NULL	NULL
30075445	2018-05-11	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 08C	Non-food contact surf...	N	5	A	2018-05-11
30075445	2017-05-18	Cycle Inspection / Initial Inspection	Violations were cited...	06D, 10F	Food contact surface...	Y	7	A	2017-05-18
30112340	2019-03-04	Cycle Inspection / Initial Inspection	Violations were cited...	02G, 10F, 10B	Cold food item held a...	Y	13	A	2019-03-04

Start with 1NF: table form, no repeating groups, PK and dependencies identified

Inspections table



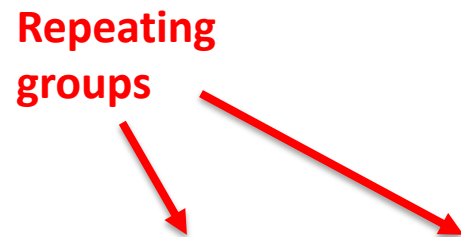
Inspections	
RestaurantID	INT(11)
InspectionDate	DATE
InspectionTypeID	INT
ActionID	INT
ViolationCode	TEXT
ViolationDescription	TEXT
CriticalFlag	VARCHAR(4)
Score	INT(11)
Grade	VARCHAR(4)
GradeDate	DATE
Indexes	

Step 1: Put in table form

Step 2: Eliminate repeating groups

- Multiple entries ViolationCode and ViolationDescription attributes because each inspection may result in multiple violations
- Remove repeating entries by making each violation its own row

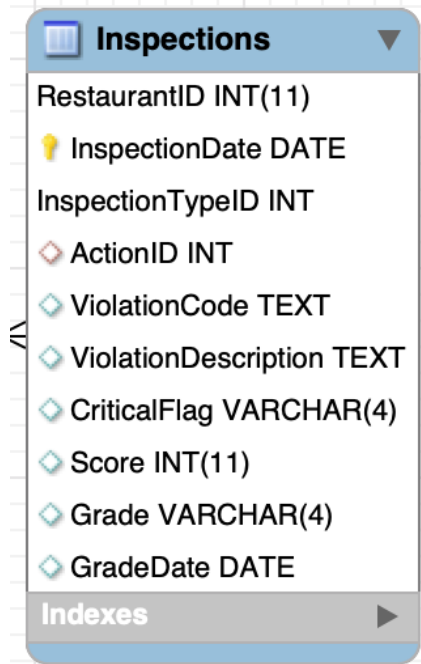
Repeating groups



Restaurant...	InspectionD...	InspectionType	ActionDescription	ViolationCode	ViolationDescription	CriticalFlag	Score	Grade	GradeDate
30075445	2019-11-04	Cycle Inspection / Re-inspection	Violations were cited...	04N	Filth flies or food/refu...	Y	7	A	2019-11-04
30075445	2019-10-21	Cycle Inspection / Initial Inspection	Violations were cited...	08A, 06D, 04L, 10B	Facility not vermin pr...	Y	17	NULL	NULL
30075445	2019-06-11	Cycle Inspection / Re-inspection	Violations were cited...	08C, 10B, 10F	Pesticide use not in a...	N	6	A	2019-06-11
30075445	2019-05-16	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 04L, 08A	Non-food contact surf...	Y	14	NULL	NULL
30075445	2018-05-11	Cycle Inspection / Initial Inspection	Violations were cited...	10F, 08C	Non-food contact surf...	N	5	A	2018-05-11
30075445	2017-05-18	Cycle Inspection / Initial Inspection	Violations were cited...	06D, 10F	Food contact surface...	Y	7	A	2017-05-18
30112340	2019-03-04	Cycle Inspection / Initial Inspection	Violations were cited...	02G, 10F, 10B	Cold food item held a...	Y	13	A	2019-03-04

Start with 1NF: table form, no repeating groups, PK and dependencies identified

Inspections table



A screenshot of a database schema tool showing the 'Inspections' table. The table has the following attributes: RestaurantID INT(11), InspectionDate DATE, InspectionTypeID INT, ActionID INT, ViolationCode TEXT, ViolationDescription TEXT, CriticalFlag VARCHAR(4), Score INT(11), Grade VARCHAR(4), and GradeDate DATE. The 'Indexes' section is visible at the bottom.

Attribute	DataType
RestaurantID	INT(11)
InspectionDate	DATE
InspectionTypeID	INT
ActionID	INT
ViolationCode	TEXT
ViolationDescription	TEXT
CriticalFlag	VARCHAR(4)
Score	INT(11)
Grade	VARCHAR(4)
GradeDate	DATE

Step 1: Put in table form

Step 2: Eliminate repeating groups

- Multiple entries ViolationCode and ViolationDescription attributes because each inspection may result in multiple violations
- Remove repeating entries by making each violation its own row

Now each
violation on
its own row

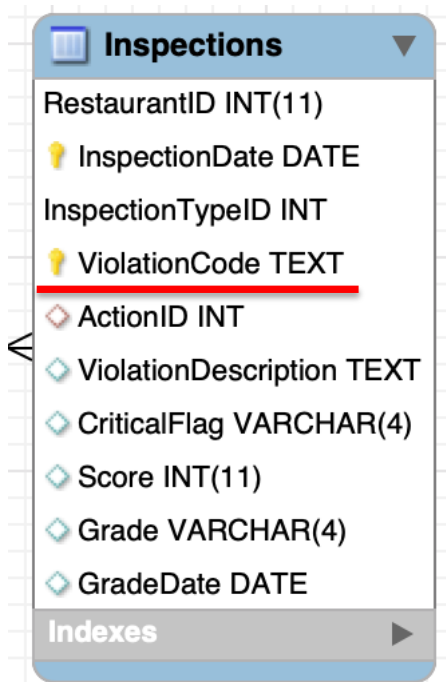


A screenshot of a database table showing inspection data. The table has columns: InspectionID, RestaurantID, InspectionDate, InspectionType, ActionDescription, ViolationCode, and ViolationDescription. A red arrow points to the ViolationCode column. The table contains 10 rows of data.

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
272957	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pro
290787	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
29149	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
305820	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	08C	Pesticide use not in accorda
119155	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
99337	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
151071	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin-proof. Ha
227147	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface im

Start with 1NF: table form, no repeating groups, PK and dependencies identified

Inspections table



Inspections	
RestaurantID	INT(11)
InspectionDate	DATE
InspectionTypeID	INT
<u>ViolationCode</u>	TEXT
ActionID	INT
ViolationDescription	TEXT
CriticalFlag	VARCHAR(4)
Score	INT(11)
Grade	VARCHAR(4)
GradeDate	DATE
Indexes	

Step 1: Put in table form

Step 2: Eliminate repeating groups

Step 3: Identify primary key

- Primary key now needs ViolationCode to uniquely identify rows
- Primary key now: RestaurantID, InspectionDate, InspectionType, ViolationCode
- Consider surrogate key
 - Four attributes becomes unwieldy
 - Surrogate key is a key whose value is assigned by the system (auto increment)

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
272957	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pro
290787	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
29149	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
305820	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	08C	Pesticide use not in accorda
119155	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
99337	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
151071	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
227147	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface im

Start with 1NF: table form, no repeating groups, PK and dependencies identified

Inspections table

InspectionID INT
RestaurantID INT(11)
InspectionDate DATE
InspectionTypeID INT
ActionID INT
ViolationCode TEXT
ViolationDescription TEXT
CriticalFlag VARCHAR(4)
Score INT(11)
Grade VARCHAR(4)
GradeDate DATE

Step 1: Put in table form

Step 2: Eliminate repeating groups

Step 3: Identify primary key

- Primary key now needs ViolationCode to uniquely identify rows
- Primary key now: RestaurantID, InspectionDate, InspectionType, ViolationCode
- Consider surrogate key
 - Four attributes becomes unwieldy
 - Surrogate key is a key assigned by the system (auto increment)

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
272957	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
290787	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
29149	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
305820	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	08C	Pesticide use not in accorda
119155	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
99337	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
151071	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
227147	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface im

Start with 1NF: table form, no repeating groups, PK and dependencies identified

Inspections table

Inspections	
InspectionID	INT
RestaurantID	INT(11)
InspectionDate	DATE
InspectionTypeID	INT
ActionID	INT
ViolationCode	TEXT
ViolationDescription	TEXT
CriticalFlag	VARCHAR(4)
Score	INT(11)
Grade	VARCHAR(4)
GradeDate	DATE

Indexes

Step 1: Put in table form

Step 2: Eliminate repeating groups

Step 3: Identify primary key

Step 4: Identify dependencies

- With surrogate key, no partial dependencies
- Transitive dependencies
 - ViolationCode → ViolationDescription, Critical Flag

Now in 1NF, move on to 2NF

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
272957	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
290787	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
29149	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
305820	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	08C	Pesticide use not in accorda
119155	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
99337	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
151071	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
227147	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface im

We will examine one table at a time, moving from First to Third Normal Form

First (1NF), Second (2NF) and Third (3NF) normal form characteristics



First Normal Form (1NF)

- Data in table format
- No repeating groups
- PK and all dependencies identified

Second Normal Form (2NF)

- 1NF plus
- No partial dependencies

Move to 2NF: remove partial dependencies

Inspections table

Inspections	
InspectionID	INT
RestaurantID	INT(11)
InspectionDate	DATE
InspectionTypeID	INT
ActionID	INT
ViolationCode	TEXT
ViolationDescription	TEXT
CriticalFlag	VARCHAR(4)
Score	INT(11)
Grade	VARCHAR(4)
GradeDate	DATE
Indexes	

Step 1: Make new tables to eliminate partial dependencies

- Partial dependency is when an attribute is dependent on only part of a composite key
- This table does not have a composite key now
 - There can be no partial dependencies
 - Table is automatically in 2NF

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
272957	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
290787	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
29149	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
305820	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	08C	Pesticide use not in accorda
119155	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface im
99337	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
151071	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
227147	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface im

Move to 2NF: remove partial dependencies

Inspections table

Inspections	
InspectionID	INT
RestaurantID	INT(11)
InspectionDate	DATE
InspectionTypeID	INT
ActionID	INT
ViolationCode	TEXT
ViolationDescription	TEXT
CriticalFlag	VARCHAR(4)
Score	INT(11)
Grade	VARCHAR(4)
GradeDate	DATE

Indexes

Step 1: Make new tables to eliminate partial dependencies

- Partial dependency is when an attribute is dependent on only part of a composite key
- This table does not have a composite key now
 - There can be no partial dependencies
 - Table is automatically in 2NF
- If we had left the primary key as RestaurantID, InspectionDate, InspectionType, ViolationCode instead of using surrogate key then:
 - ViolationDescription and CriticalFlag are partial dependencies (depend only on ViolationCode)
 - We would deal with that at this stage by creating a new table and reassigning dependent attributes
 - With the surrogate key, we deal with it in 3NF

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
1	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
2	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface in
3	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
4	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface in
5	30075445	2019-10-21	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
6	30075445	2019-11-04	Cycle Inspection / R...	Violations were cited in...	04N	Filth flies or food/refuse/co

We will examine one table at a time, moving from First to Third Normal Form

First (1NF), Second (2NF) and Third (3NF) normal form characteristics



First Normal Form (1NF)

- Data in table format
- No repeating groups
- PK and all dependencies identified

Second Normal Form (2NF)

- 1NF plus
- No partial dependencies

Third Normal Form (3NF)

- 2NF plus
- No transitive dependencies

Move to 3NF: remove transitive

Inspections table

Inspections	
InspectionID	INT
RestaurantID	INT(11)
InspectionDate	DATE
InspectionTypeID	INT
ActionID	INT
ViolationCode	TEXT
ViolationDescription	TEXT
CriticalFlag	VARCHAR(4)
Score	INT(11)
Grade	VARCHAR(4)
GradeDate	DATE

Indexes

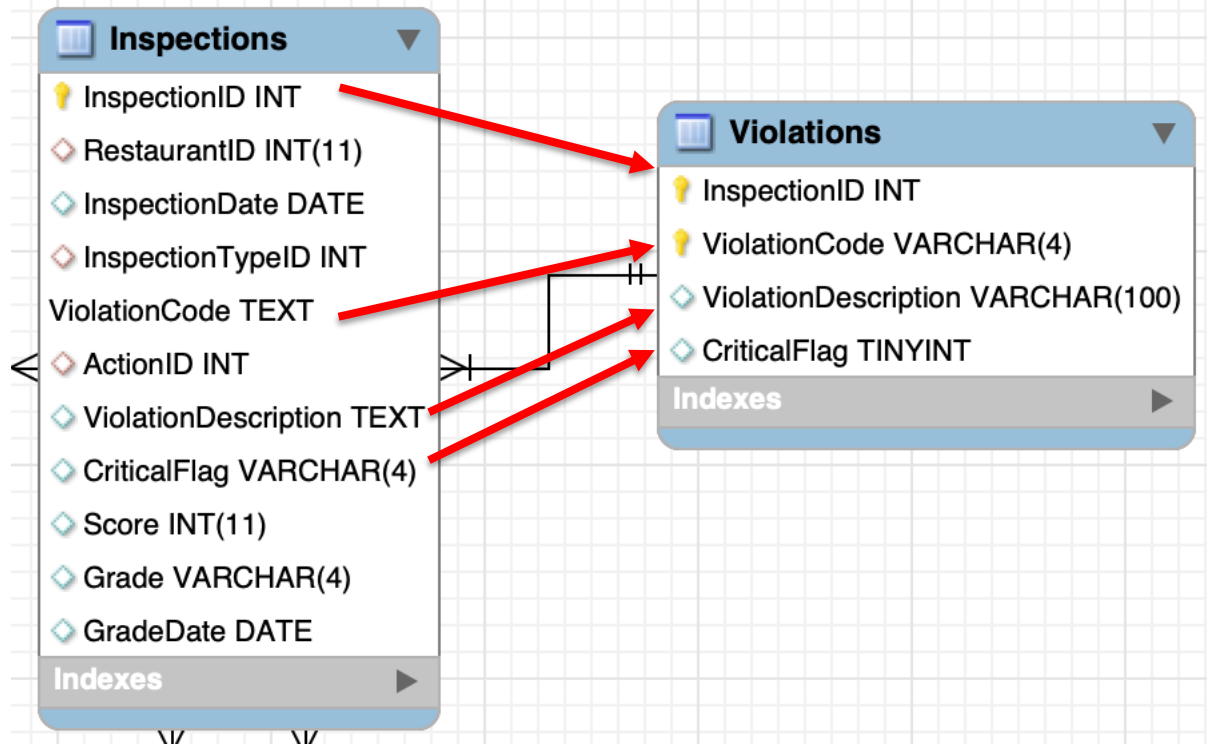
Step 1: Make new tables to eliminate transitive dependencies

- Transitive dependency: If $A \rightarrow B$ and $B \rightarrow C$, then $A \rightarrow C$
- Identify by looking for dependencies on nonprime attributes
- $\text{ViolationCode} \rightarrow \text{ViolationDescription, Critical Flag}$
- Make new tables to eliminate transitive dependencies
 - Create table for ViolationCodes with ViolationCode as PK and ViolationDescription and CriticalFlag as attributes
 - Make foreign key entry in Inspections table for ViolationCode

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
1	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
2	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface in
3	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
4	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface in
5	30075445	2019-10-21	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
6	30075445	2019-11-04	Cycle Inspection / R...	Violations were cited in...	04N	Filth flies or food/refuse/co

Move dependent attributes into their own table

Inspections table



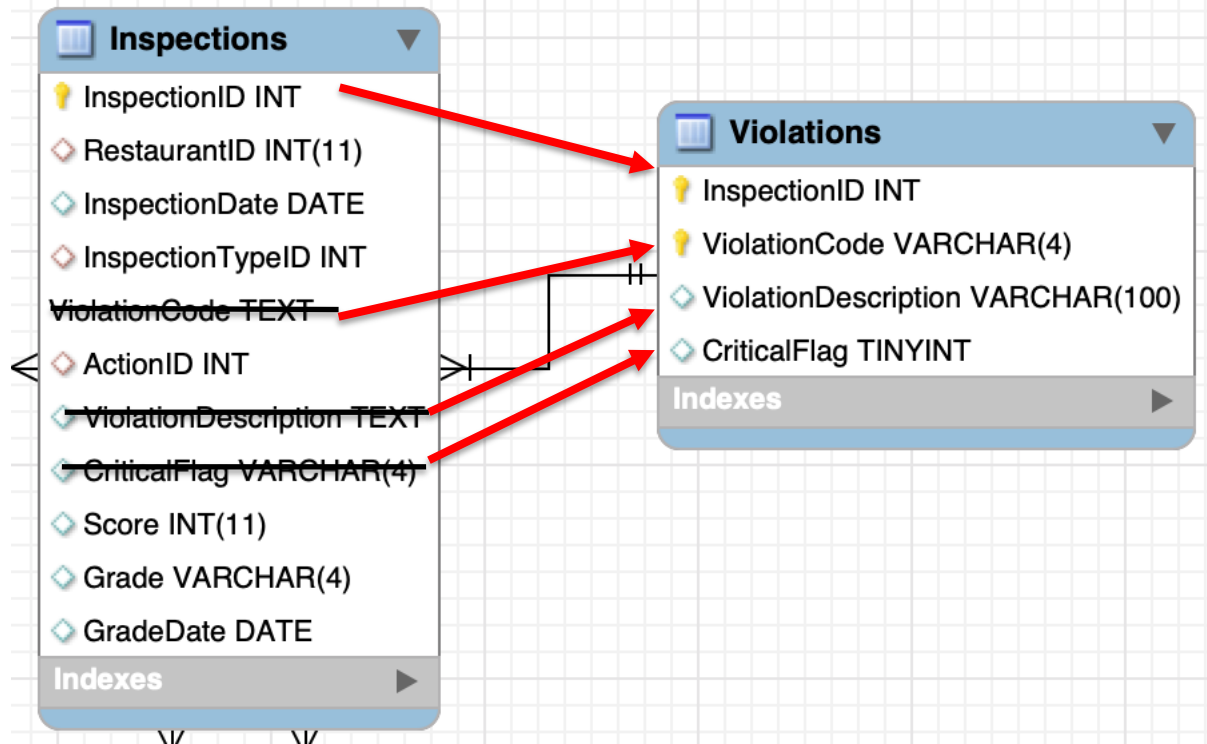
Move dependent attributes into their own table (we would have done this with partial dependencies also)

Use PK in new table as FK in Inspections

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
1	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
2	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface in
3	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
4	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface in
5	30075445	2019-10-21	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
6	30075445	2019-11-04	Cycle Inspection / R...	Violations were cited in...	04N	Filth flies or food/refuse/co

Move dependent attributes into their own table

Inspections table



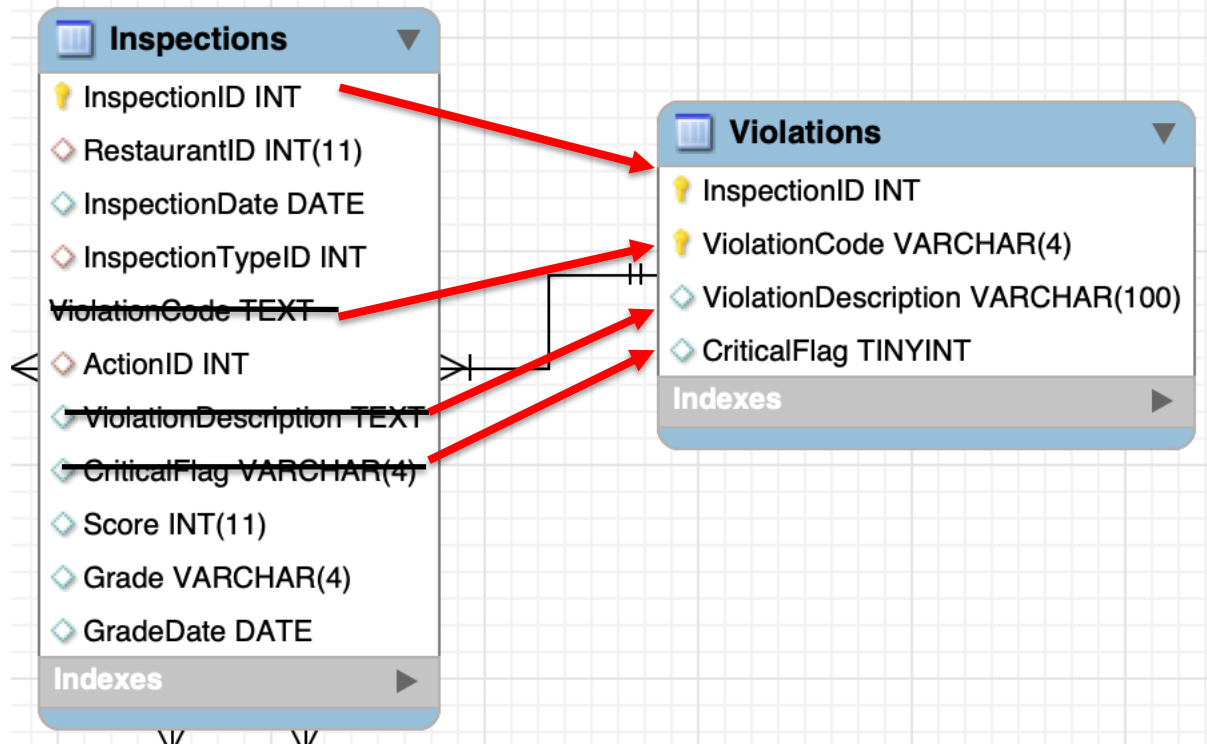
Move dependent attributes into their own table (we would have done this with partial dependencies also)

Use PK in new table as FK in Inspections

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
1	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
2	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface in
3	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
4	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface in
5	30075445	2019-10-21	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
6	30075445	2019-11-04	Cycle Inspection / R...	Violations were cited in...	04N	Filth flies or food/refuse/co

Move dependent attributes into their own table

Inspections table



Move dependent attributes into their own table (we would have done this with partial dependencies also)

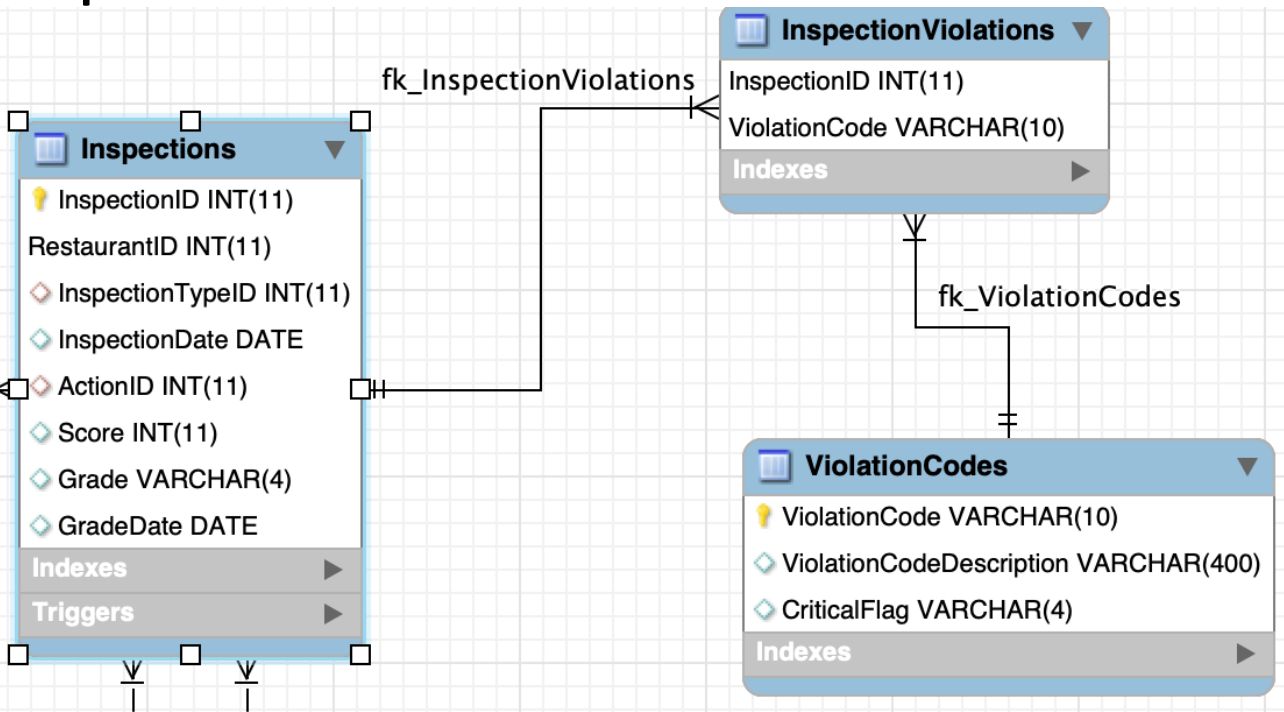
Use PK in new table as FK in Inspections

Problem! There is a M:N relationship between Inspections and InspectionCodes

InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
1	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
2	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface in
3	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
4	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface in
5	30075445	2019-10-21	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
6	30075445	2019-11-04	Cycle Inspection / R...	Violations were cited in...	04N	Filth flies or food/refuse/co

Use a joining table for M:N relationships

Inspections table



Use a joining table

Inspection table get smaller

- **New tables created**
- **Dependent attributes moved into new tables**

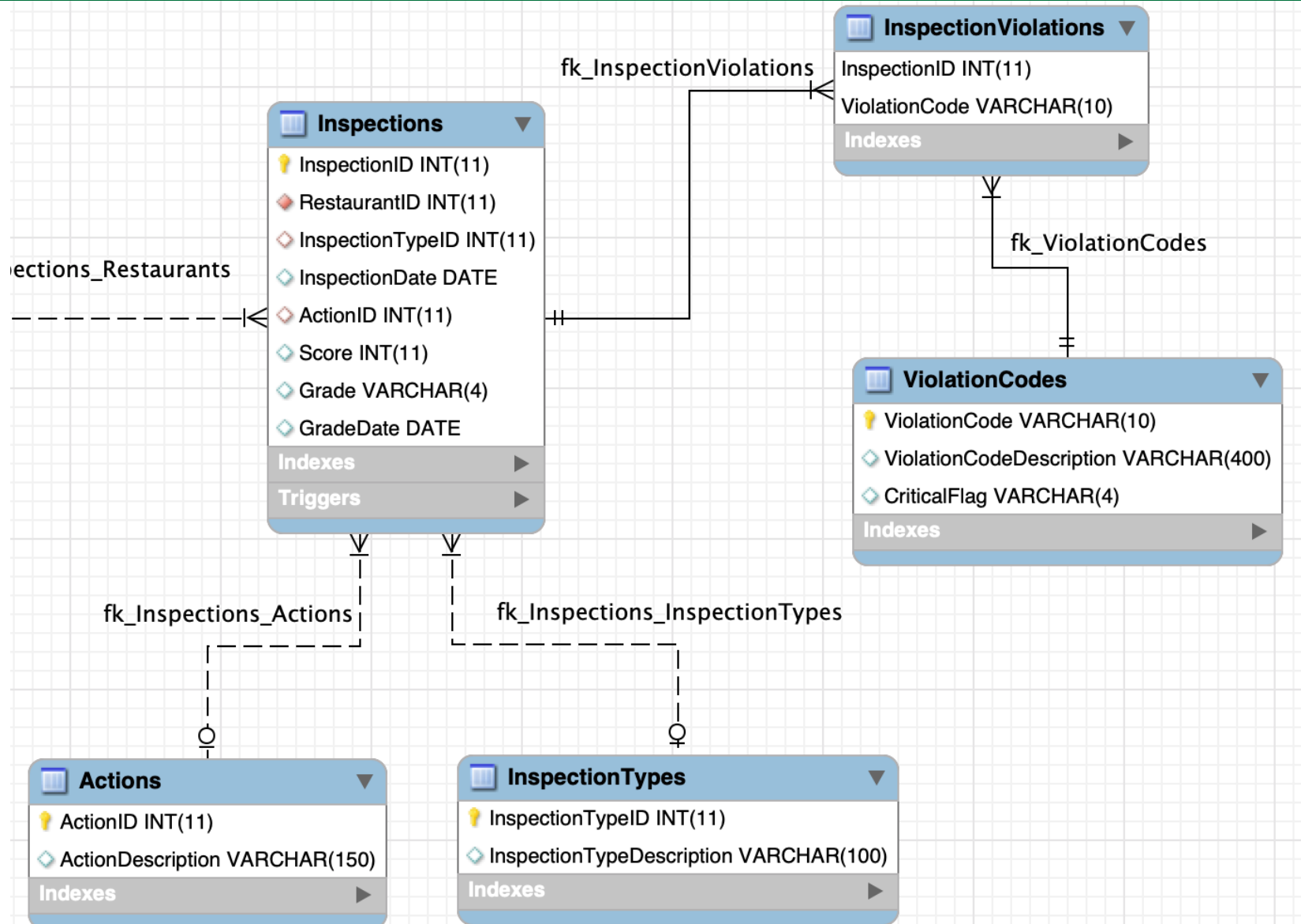
InspectionID	RestaurantID	InspectionDate	InspectionType	ActionDescription	ViolationCode	ViolationDescription
1	30075445	2017-05-18	Cycle Inspection / Ini...	Violations were cited in...	06D	Food contact surface not pr
2	30075445	2018-05-11	Cycle Inspection / Ini...	Violations were cited in...	10F	Non-food contact surface in
3	30075445	2019-05-16	Cycle Inspection / Ini...	Violations were cited in...	04L	Evidence of mice or live mic
4	30075445	2019-06-11	Cycle Inspection / R...	Violations were cited in...	10F	Non-food contact surface in
5	30075445	2019-10-21	Cycle Inspection / Ini...	Violations were cited in...	08A	Facility not vermin proof. Ha
6	30075445	2019-11-04	Cycle Inspection / R...	Violations were cited in...	04N	Filth flies or food/refuse/co

Normalization is valuable because it helps eliminate data redundancies

Other steps to consider after reaching 3NF

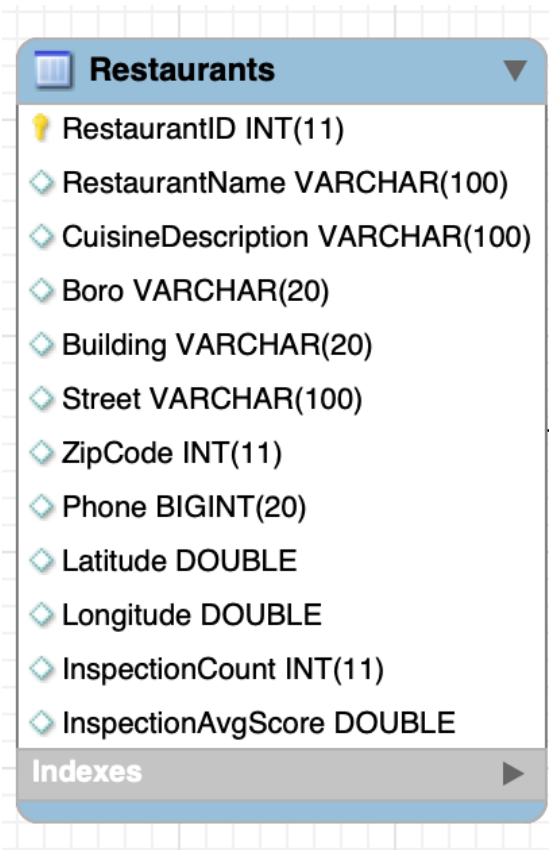
- Identify new attributes and new relationships (did we forget anything?)
- Refine attribute atomicity (will we ever need part of an attribute like first name if storing name as first and last name)
 - Atomic attribute: cannot be further subdivided
 - Atomicity: characteristic of an atomic attribute
- Evaluate using derived vs. stored attributes
- Consider foreign key requirements
 - Here we want Actions and InspectionTypes to be selected from a set of known values
 - Create new tables for these, use PK as FK in Inspections

Normalized Inspections table



Sometimes we choose to denormalize

Restaurants table



A screenshot of a database table definition for a table named 'Restaurants'. The table is displayed in a window with a blue header bar. The table has 12 columns, each with a data type and length. The columns are: RestaurantID (INT(11)), RestaurantName (VARCHAR(100)), CuisineDescription (VARCHAR(100)), Boro (VARCHAR(20)), Building (VARCHAR(20)), Street (VARCHAR(100)), ZipCode (INT(11)), Phone (BIGINT(20)), Latitude (DOUBLE), Longitude (DOUBLE), InspectionCount (INT(11)), and InspectionAvgScore (DOUBLE). The table is shown with a grid background.

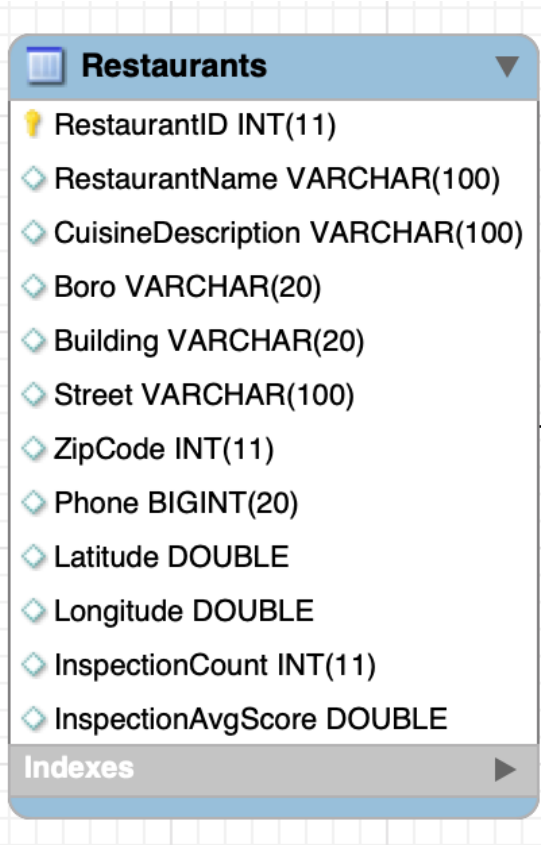
Restaurants
RestaurantID INT(11)
RestaurantName VARCHAR(100)
CuisineDescription VARCHAR(100)
Boro VARCHAR(20)
Building VARCHAR(20)
Street VARCHAR(100)
ZipCode INT(11)
Phone BIGINT(20)
Latitude DOUBLE
Longitude DOUBLE
InspectionCount INT(11)
InspectionAvgScore DOUBLE

Can you identify any dependencies in the Restaurants table?

- Can there be any partial dependencies here?
- Can there be any transitive dependencies?

Sometimes we choose to denormalize

Restaurants table



A screenshot of a database schema viewer showing the 'Restaurants' table. The table has 12 columns: RestaurantID (INT(11), primary key), RestaurantName (VARCHAR(100)), CuisineDescription (VARCHAR(100)), Boro (VARCHAR(20)), Building (VARCHAR(20)), Street (VARCHAR(100)), ZipCode (INT(11)), Phone (BIGINT(20)), Latitude (DOUBLE), Longitude (DOUBLE), InspectionCount (INT(11)), and InspectionAvgScore (DOUBLE). Below the table list is a tab labeled 'Indexes'.

Column Name	Data Type
RestaurantID	INT(11)
RestaurantName	VARCHAR(100)
CuisineDescription	VARCHAR(100)
Boro	VARCHAR(20)
Building	VARCHAR(20)
Street	VARCHAR(100)
ZipCode	INT(11)
Phone	BIGINT(20)
Latitude	DOUBLE
Longitude	DOUBLE
InspectionCount	INT(11)
InspectionAvgScore	DOUBLE

Can you identify any dependencies in the Restaurants table?

- Can there be any partial dependencies here? **No: key is single attribute**
- Can there be any transitive dependencies?

Sometimes we choose to denormalize

Restaurants table



A screenshot of a database table definition for 'Restaurants'. The table has the following columns:

Column Name	Column Type
RestaurantID	INT(11)
RestaurantName	VARCHAR(100)
CuisineDescription	VARCHAR(100)
Boro	VARCHAR(20)
Building	VARCHAR(20)
Street	VARCHAR(100)
ZipCode	INT(11)
Phone	BIGINT(20)
Latitude	DOUBLE
Longitude	DOUBLE
InspectionCount	INT(11)
InspectionAvgScore	DOUBLE

Below the columns, there is a section for 'Indexes' with a right-pointing arrow.

Can you identify any dependencies in the Restaurants table?

- Can there be any partial dependencies here? **No: key is single attribute**
- Can there be any transitive dependencies? **Yes: ZipCode gives Boro**

Could make a table with ZipCode as key and Boro as attribute and look up Boro with JOIN as needed

Sometimes we choose to denormalize

Restaurants table



A screenshot of a database schema tool showing the structure of the 'Restaurants' table. The table has 12 columns: RestaurantID (INT(11), primary key), RestaurantName (VARCHAR(100)), CuisineDescription (VARCHAR(100)), Boro (VARCHAR(20)), Building (VARCHAR(20)), Street (VARCHAR(100)), ZipCode (INT(11)), Phone (BIGINT(20)), Latitude (DOUBLE), Longitude (DOUBLE), InspectionCount (INT(11)), and InspectionAvgScore (DOUBLE). The 'Indexes' tab is selected at the bottom.

Column Name	Data Type
RestaurantID	INT(11)
RestaurantName	VARCHAR(100)
CuisineDescription	VARCHAR(100)
Boro	VARCHAR(20)
Building	VARCHAR(20)
Street	VARCHAR(100)
ZipCode	INT(11)
Phone	BIGINT(20)
Latitude	DOUBLE
Longitude	DOUBLE
InspectionCount	INT(11)
InspectionAvgScore	DOUBLE

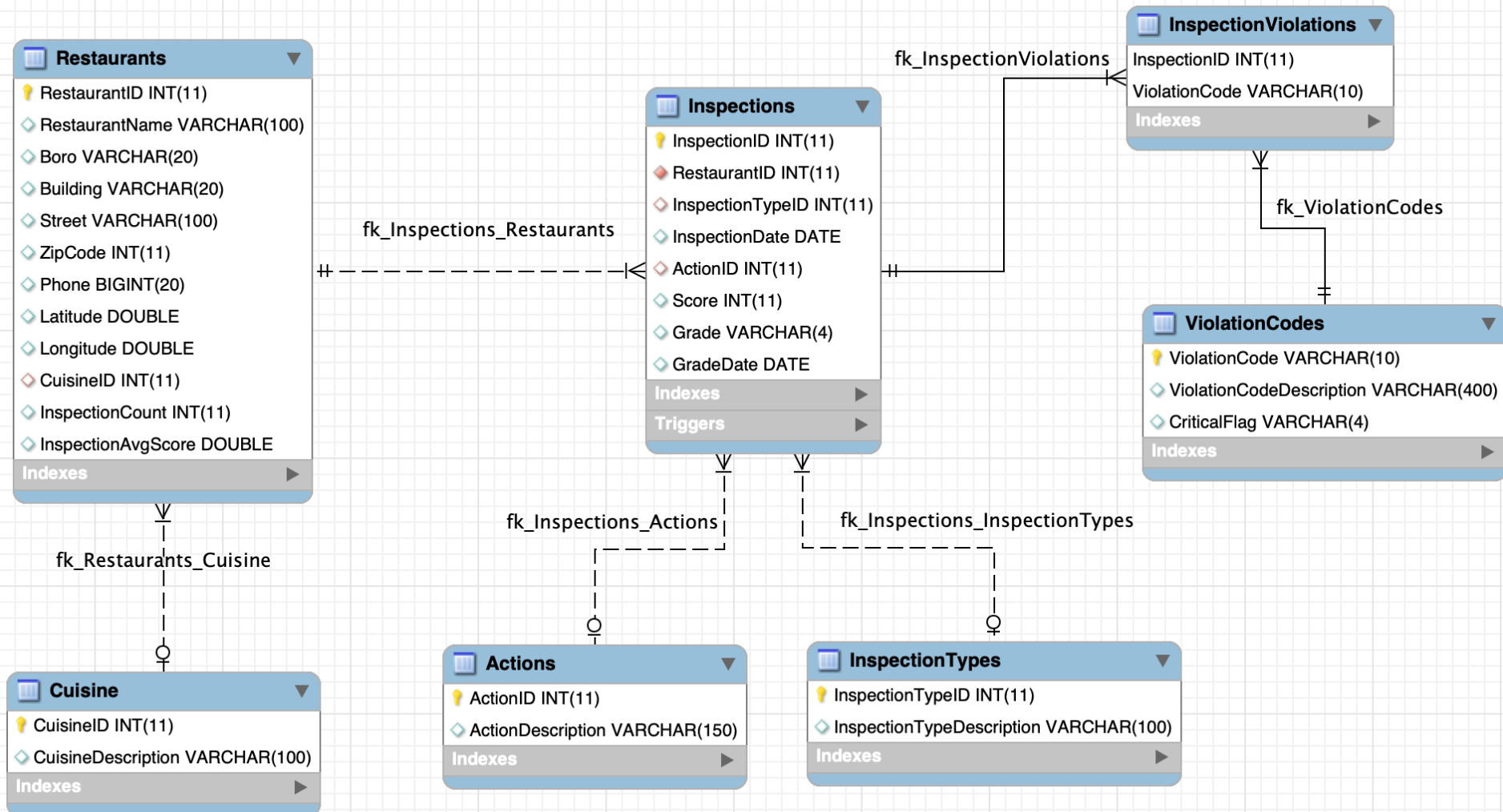
Can you identify any dependencies in the Restaurants table?

- Can there be any partial dependencies here? **No: key is single attribute**
- Can there be any transitive dependencies? **Yes: ZipCode gives Boro**

For simplicity we choose to keep this table denormalized (2NF, has transitive dependency)

But, want Cuisine to be selected from a small number of options, so make a Cuisine table and use Cuisine as FK in Restaurants table

Final normalized design



Normal forms

Normalization: evaluating and correcting table structures to minimize data redundancies

Normal Form	Characteristic
First normal form (1NF)	Table format, no repeating groups, PK and dependencies identified
Second normal form (2NF)	1NF and no partial dependencies
Third normal form (3NF)	2NF and no transitive dependencies
Boyce-Codd normal form (BCNF)	Every determinant is a candidate key (special case of 3NF)
Fourth normal form (4NF)	3NF and no independent multivalued dependencies
Fifth normal form (5NF) and more	Mainly of academic interest only

**Normally
good enough**



