Data Management and Database Design

DMDD 6210

Week #3

Northeastern University

What had we learnt till now?

DATA MODELING

E.F CODD RULES

ENTITY AND ATTRIBUTES

RELATIONSHIPS

This week….

- Database environment setup Demo
  - User
  - Schema
  - Grants
  - Permissions
- Normalization
- SQL Introduction

Keys

- In Relational model Keys are important as –
  - 🔧It allows us to uniquely identify a tuple
  - 🔧It can be a combination of one or more attributes

Keys

- In Relational model Keys are important as –
  - 🔧It allows us to uniquely identify a tuple
  - 🔧It can be a combination of one or more attributes

Different types of keys -

- 🔑*Natural Key*
- 🔑*Super Key*
- 🔑*Candidate Key*
- 🔑*Primary Key*

Natural Key

- Natural keys is –
  - An Identifier which can be used to uniquely identify data
  - **Example**: **Invoice number, Registration number, Part number, etc.,**
- If an entity has Natural identifier –
  - We can use it as **Primary key** for that respective **entity**
  - Most of the natural keys are acceptable as primary key identifiers

Natural Key

- Natural keys is –
  - An Identifier which can be used to uniquely identify data
  - **Example**: **Invoice number, Registration number, Part number, etc.,**
- If an entity has Natural identifier –
  - We can use it as **Primary key** for the respective **entity**
  - Most of the natural keys are acceptable as primary key identifiers

- **Foreign key (Reference key)** is an attribute that is mapped/linked to another table's Primary key

Super Key

1 or more attributes, that uniquely identifies a tuple in a relation

Example: Entity **Product** (Product_Id, Product_Name, Manufacturer)

- Product_Id
- Product_Id, ProductName
- Product_Id, ProductName, Manufacturer
- Product_Id, Manufacturer
- Product_Name, Manufacturer

*Each **super key** can **uniquely** identify each **tuple***

Candidate key

Candidate key is a super key with no redundancy

- A proper subset of a super key (*Least number of columns*)
- The **set of candidate keys** form the base for selection of a single **primary key**

Example: Entity **Product** (Product_Id, Product_Name, Manufacturer)

- Product_Id
- Product_Id, ProductName
- Product_Id, ProductName, Manufacturer
- Product_Id, Manufacturer
- Product_Name, Manufacturer

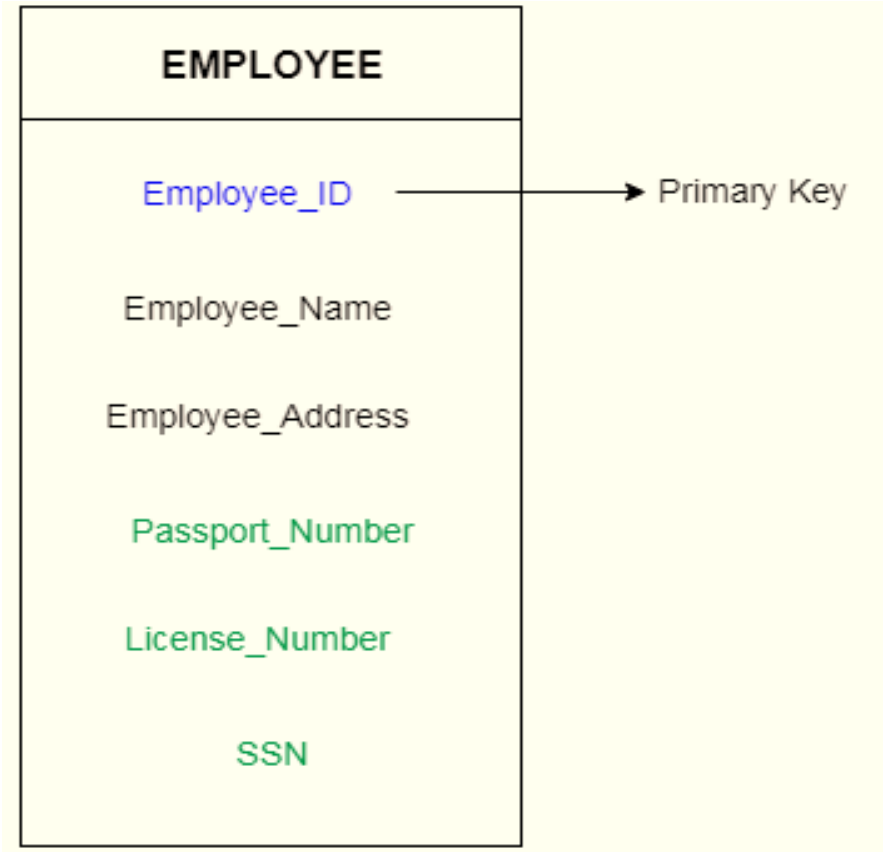
Product_id is eligible for Candidate Key

Primary key

- **Characteristics**
  - Unique values
  - No change over a ***period of time***
  - Non intelligent
  - Minimum number of Attributes (Recommended One attribute)
  - Numeric Datatype
  - Not part of any PII data

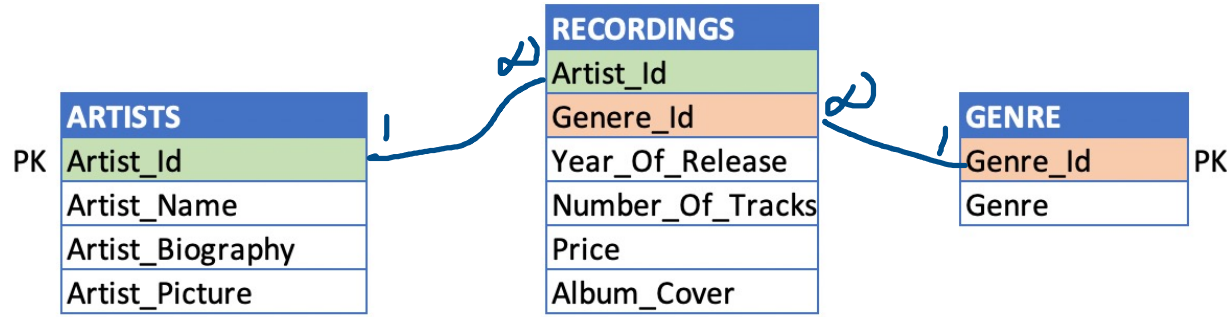
From the below table which column is best fit for a Primary Key?

EMPLOYEE
EMP_ID
EMP_NAME
EMP_ADDRESS
PASSPORT_NUMBER
LICENSE_NUMBER
SSN



Composite Primary key

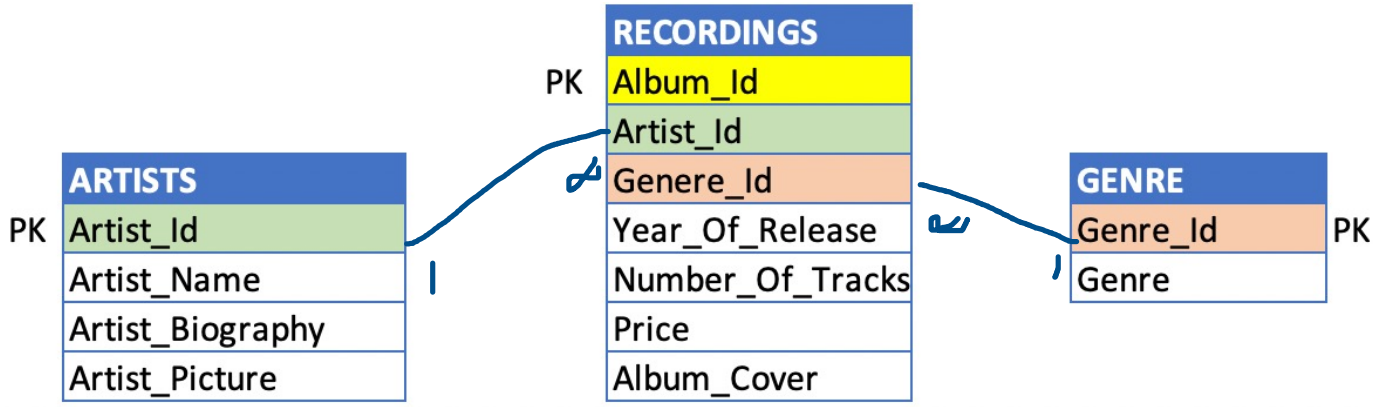
- A **composite Primary key** is combination of more than one **key** that uniquely identifies each tuple
- Combination of each Primary key is allowed only once



Artist_Id and Genere_Id Composite Primary Key

Surrogate key

- Surrogate key is created in case where a table has more than one natural key
- Table has more than one column to identify unique row
- Its not part of application data
- Usually, these values are auto generated

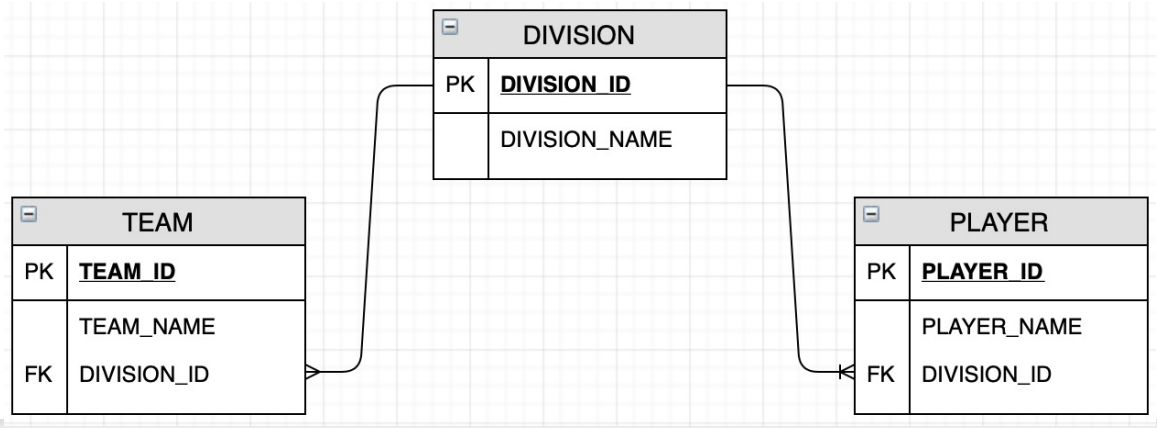


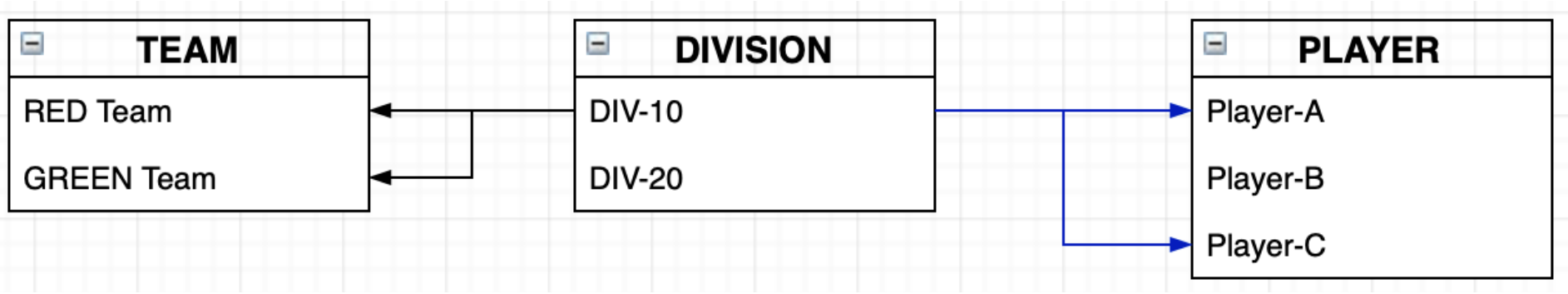
Exercise

- **Requirement**
  - Basketball league has many divisions
  - Division has many players
  - Division has many teams

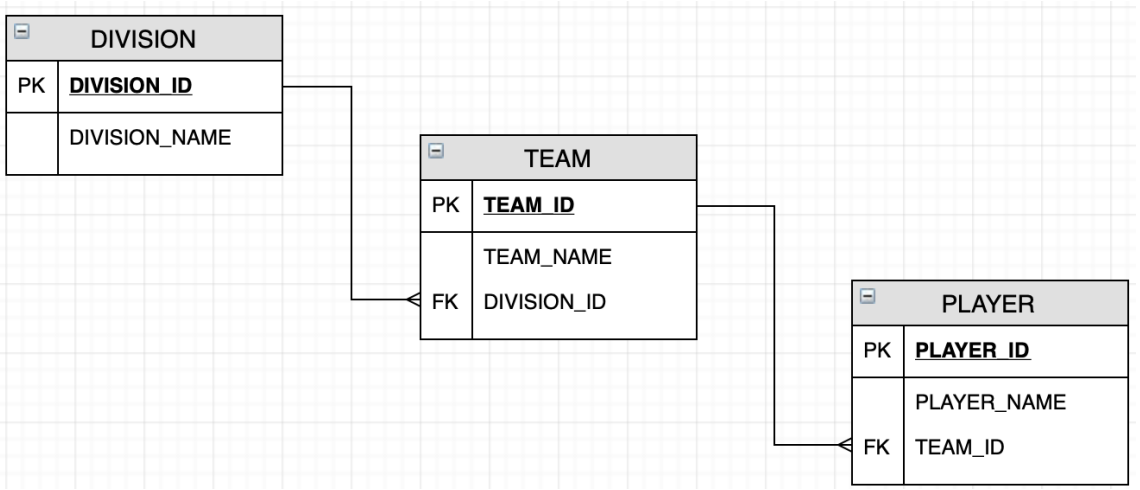
Design the model

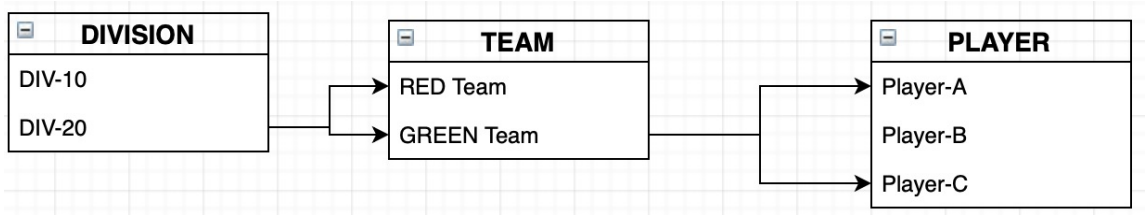
Will this works?





Solution


DIVISION
PK
DIVISION_ID
DIVISION_NAME
TEAM
PK
TEAM_ID
TEAM_NAME
FK
DIVISION_ID
PLAYER
PK
PLAYER_ID
PLAYER_NAME
FK
TEAM_ID


DIVISION
DIV-10
DIV-20
TEAM
RED Team
GREEN Team
PLAYER
Player-A
Player-B
Player-C

Normalization

Normalization

- Normalization is a –
  - Process of refining data model build by **ER diagram**
  - Design technique that begins by examining Relationships between attributes

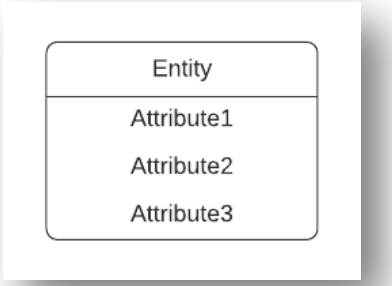
Normalization

- Normalization is a –
  - Process of refining data model build by **ER diagram**
  - Design technique that begins by examining Relationships between attributes to –
    - Ensures minimum redundancy of data
    - Easier for users to access data
    - Easy to maintain
    - Takes minimum space to store data
    - Reduces anomalies
      - Insert, Update and Delete should affect only desired ROW

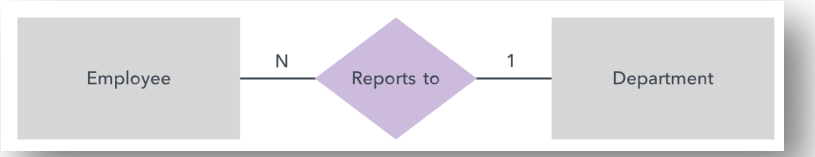
ERD – ER Diagram

- **What is ER Diagram (ERD)?**
  - Model and design relational databases, in terms of logic and business rules
  - This is the first step that happens for database dependent applications
  - Helps to troubleshoot existing problems

- **Components of ERD –**
  - Entity – A definable object such as customer, student, product and shown as Rectangle
  - Relationship – How entities connect each other and show as diamond
  - Attribute – Characteristics of an Entity and show as oval or sometimes circle.



- **Limitations –**
  - Only for Relational DBs (Structured)
  - Cannot be applied on unstructured



ERD – ER Diagram

- **Tools** –
  - Erwin
  - Toad Data modeler
  - SQL Data modeler
  - Navicat data modeler
  - E/R Studio
- **Generating**
  - Design based on code is Reverse Engineering
  - Code based on Design is Forward Engineering

Dependencies

- Functional
- Transitive
- Partial

Functional Dependencies

Important concept associated with Normalization is **Functional Dependencies**

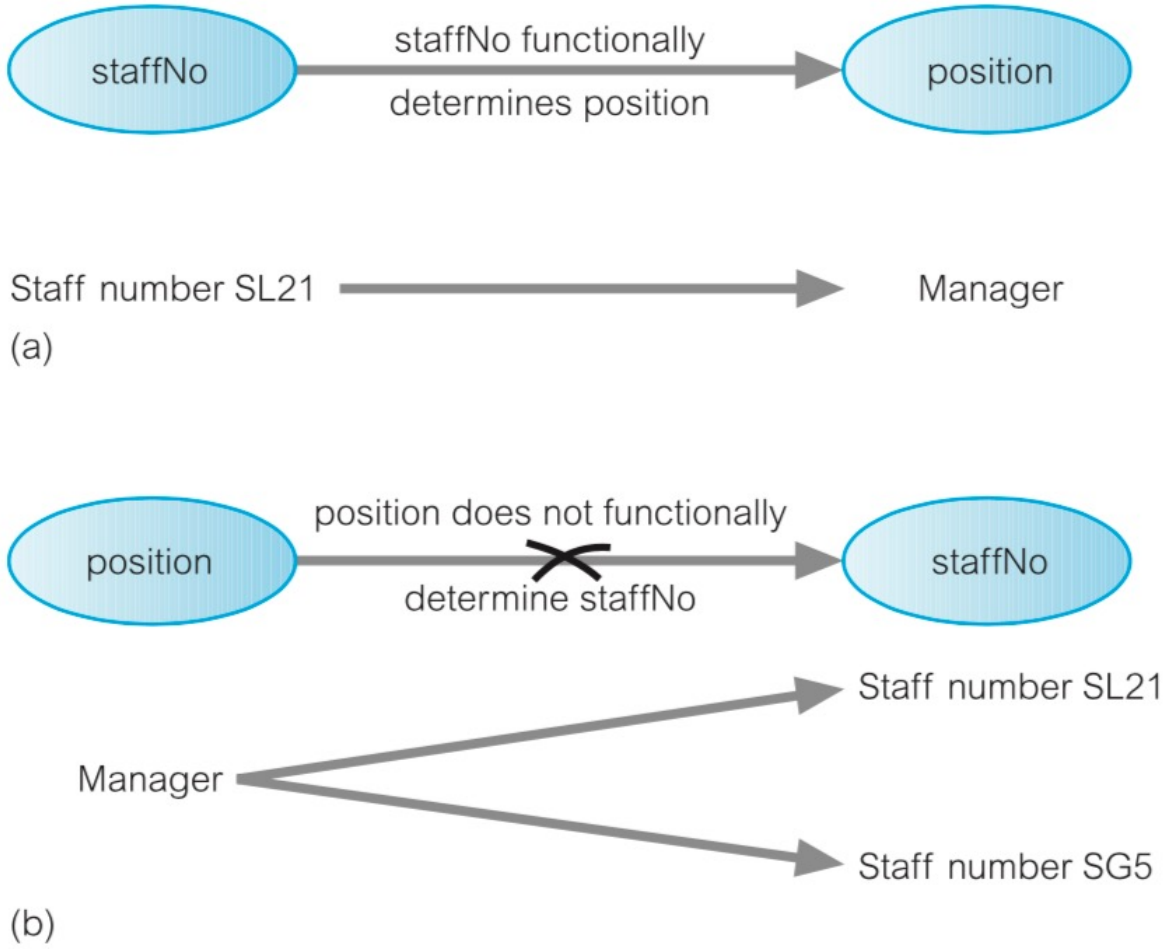
- The attribute *B* is fully functionally dependent on the attribute *A,* if each value of *A* determines one and only one value of *B*

- Example: Product_ID → Product_Name
  - *Product ID determines Product Name*
    - *Product Name is **Dependent** Attribute*
    - *Product ID is **Determinant***

  - *We can also say Product_ID **functionally determines** Product_Name*

Functional Dependencies

- **Another Example**



Partial Dependency

- A functional dependency A → B is called **partial** dependency if –
  - There is an attribute that can be removed from **A** and yet the dependency still holds

Partial Dependency is similar to functional dependency when non-prime-key attribute(s) are functionally dependent on a **part of the Candidate key**.

Student_ID	Student_fName	Student_lname	Course_ID	Course_name
S001	John	Smith	C8899	Artificial Intelligence
S002	Brad	Smith	C8899	Artificial Intelligence

Student ID + Course ID is unique – Primary key

Course Name is dependent on Course ID and doesn't require Student ID

To resolve this, we create a parent table called course and reference to student.

Transitive Dependency

- An indirect relationship between data elements
- A, B, and C are attributes of a relation such that if
  - A → B and B → C, then **C** is transitively dependent on **A** via **B**
- If we know Model, we can get Manufacturer, Using Manufacturer we can get phone number
  - **Model** –> **contact** is transitive dependent

- **Problems if we have Transitive dependency –**
  - Cannot add Manufacturer without a Model
  - If we delete Kia and Sonata models, then Manufacturer and his contact is removed
  - Adding additional attributes of Manufacture adds repetition
  - Update leads more than one row modification

Model	Manufacturer	Contact#
Camry	Toyota	9819876677
Corella	Toyota	9819876677
Accord	Honda	8889997766
Civic	Honda	8889997766
Acura	Honda	8889997766
Kia	Hyundai	8001112222
Sonata	Hyundai	8001112222

Normalization

Normalization

1st Normal Form

2nd Normal Form

3rd Normal Form

Boyce-Codd normal form (BCNF)

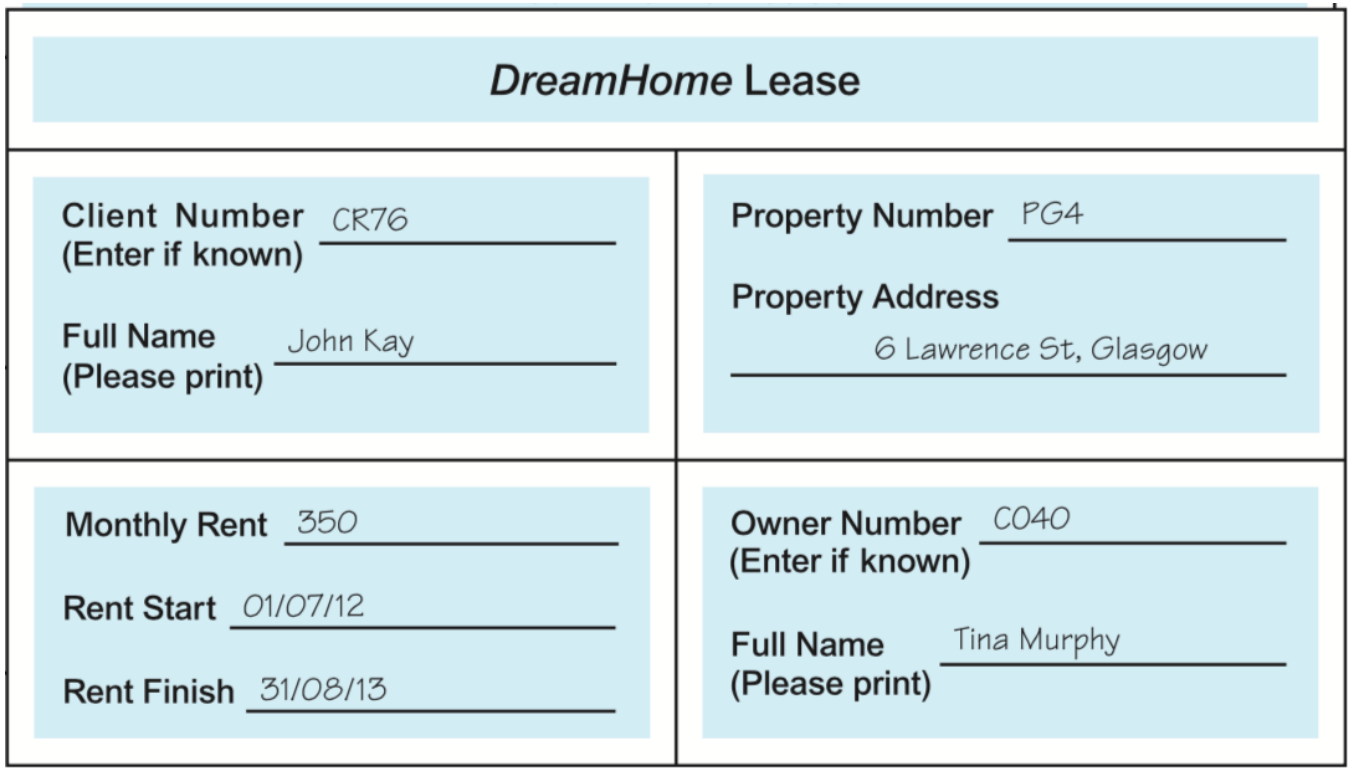
Fourth normal form (4NF)

First Normal form – 1NF

- **Steps –**
  - Identify **repeating** groups of fields
  - **Remove** repeating groups to separate table
  - Identify **keys** for the tables
  - Key of parent table is brought as part of **concatenated key** to the second table

Raw Data

- **Assumption** –
  - Client rents a given property only once
  - Cannot rent more than one property at a time



DreamHome Lease

Client Number (Enter if known) CR76

Full Name (Please print) John Kay

Property Number PG4

Property Address 6 Lawrence St, Glasgow

Monthly Rent 350

Rent Start 01/07/12

Rent Finish 31/08/13

Owner Number (Enter if known) C040

Full Name (Please print) Tina Murphy

First Normal form – 1NF

CLIENT RENTAL

Client_No	Client_Name	Property_No	Address	Lease_Start_Dt	Lease_End_Dt	Rent	Owner_No	Owner_Name
C112	Scott	P4 P5	1 Main St, Burbank, CA 6 Nova Dr, Grafton, MA	1-Jul-2012 1-Sep-2013	31-Aug-2013 1-Sep-2014	350 800	CO40 CO93	Tina Tony
C113	Sam	P4 P5 P6	1 Main St, Burbank, CA 6 Nova Dr, Grafton, MA 12 Village Way, Calbas, CA	1-Sep-2011 10-Oct-2012 1-Nov-2014	10-Jun-2012 1-Dec-2013 10-Aug-2015	350 800 450	CO40 CO93 CO93	Tina Tony Tony

First Normal form – 1NF

CLIENT RENTAL

Client_No	Client_Name	Property_No	Address	Lease_Start_Dt	Lease_End_Dt	Rent	Owner_No	Owner_Name
C112	Scott	P4	1 Main St, Burbank, CA	1-Jul-2012	31-Aug-2013	350	CO40	Tina
		P5	6 Nova Dr, Grafton, MA	1-Sep-2013	1-Sep-2014	800	CO93	Tony
C113	Sam	P4	1 Main St, Burbank, CA	1-Sep-2011	10-Jun-2012	350	CO40	Tina
		P5	6 Nova Dr, Grafton, MA	10-Oct-2012	1-Dec-2013	800	CO93	Tony
		P6	12 Village Way, Calbas, CA	1-Nov-2014	10-Aug-2015	450	CO93	Tony

#1 **Lets create a table appropriately with data into each row (Intersection of an attribute should be single value**

Client_No	Client_Name	Property_No	Address	Lease_Start_Dt	Lease_End_Dt	Rent	Owner_No	Owner_Name
C112	Scott	P4	1 Main St, Burbank, CA	1-Jul-2012	31-Aug-2013	350	CO40	Tina
C112	Scott	P5	6 Nova Dr, Grafton, MA	1-Sep-2013	1-Sep-2014	800	CO93	Tony

The relation contains data describing clients, property rented, and property owners repeated many times

Which results in redundancy

First Normal form – 1NF

#2 **Repeating Group = (property_No, Address, Lease_Start_Dt, Lease_End_Dt, rent, owner_No, owner_Name) removal**

CLIENT

Client_No	Client_Name
C112	Scott
C112	Scott

PROPERTY_RENTAL_OWNER

Client_No	Property_No	Address	Lease_Start_Dt	Lease_End_Dt	Rent	Owner_No	Owner_Nam
C112	P4	1 Main St, Burbank, CA	1-Jul-2012	31-Aug-2013	350	CO40	Tina
C112	P5	6 Nova Dr, Grafton, MA	1-Sep-2013	1-Sep-2014	800	CO93	Tony
C113	P4	1 Main St, Burbank, CA	1-Sep-2011	10-Jun-2012	350	CO40	Tina
C113	P5	6 Nova Dr, Grafton, MA	10-Oct-2012	1-Dec-2013	800	CO93	Tony
C113	P6	12 Village Way, Calbas, CA	1-Nov-2014	10-Aug-2015	450	CO93	Tony

Second Normal form – 2NF

- **Steps –**
  - Check if all fields are dependent on the whole key
  - Remove fields that depend on part of the key
  - Group partially dependent fields as a sequence table
  - Name the tables
  - Identify keys to the tables

Second Normal form – 2NF

CLIENT

Client_No	Client_Name
C112	Scott
C113	Sam

PEOPERTY_ONWER

Property_No	Address	Rent	O
P4	1 Main St, Burbank, CA	350	C
P5	6 Nova Dr, Grafton, MA	800	C
P4	1 Main St, Burbank, CA	350	C
P5	6 Nova Dr, Grafton, MA	800	C
P6	12 Village Way, Calbas, CA	450	C

RENTAL

Client_No	Property_No	Lease_Start_Dt	Lease_End_Dt
C112	P4	1-Jul-2012	31-Aug-2013
C112	P5	1-Sep-2013	1-Sep-2014
C113	P4	1-Sep-2011	10-Jun-2012
C113	P5	10-Oct-2012	1-Dec-2013
C113	P6	1-Nov-2014	10-Aug-2015

- Concept of 2NF is based on full functional dependency
- Functional dependencies are –
  - Client_No, Property_No, Lease_Start_Dt, Lease_End_Dt (Primary Key)
    - As Client can rent same property again as 2nd lease so we need Start and End dates to the key
  - Property_No, Address, Rent, Owner_No, Owner_Name (Partial Dependency)

Third Normal form – 3NF

- **Remove Fields that –**
  - **Depend on other non-key fields (Transitive dependency)**
  - **Can be calculated or derived**

- Group interdependent fields as separate tables.
- Identify such tables and name them with appropriate key.

Third Normal form – 3NF

- **Lets do functional dependency for Client, Rental and Property owners –**
  - **CLIENT**
    - Client_No, Client_Name
  - **RENTAL**
    - Client_No, Property_No, Lease_Start_Dt, Lease_End_Dt
  - **PROPERTY_OWNER**
    - Property_No, Address, Rent, Owner_No, Owner_Name

Third Normal form – 3NF

CLIENT

Client_No	Client_Name
C112	Scott
C113	Sam

PRPOERTY

Property_No	Address	Rent	Owner_No
P4	1 Main St, Burbank, CA	350	CO40
P5	6 Nova Dr, Grafton, MA	800	CO93
P4	1 Main St, Burbank, CA	350	CO40
P5	6 Nova Dr, Grafton, MA	800	CO93
P6	12 Village Way, Calbas, CA	450	CO93

RENTAL

Client_No	Property_No	Lease_Start_Dt	Lease_End_Dt
C112	P4	1-Jul-2012	31-Aug-2013
C112	P5	1-Sep-2013	1-Sep-2014
C113	P4	1-Sep-2011	10-Jun-2012
C113	P5	10-Oct-2012	1-Dec-2013
C113	P6	1-Nov-2014	10-Aug-2015

Surrogate key

OWNER

Owner_No	Owner_Name
CO40	Tina
CO93	Tony
CO40	Tina
CO93	Tony
CO93	Tony

Another example

Name	Address	Redbox Movie Rented	Actor_Name
Tony	1 north Ave, Boston, MA 02112	Clash of titans	Matt
Tony	1 north Ave, Boston, MA 02112	Pirates of Caribbean	John
Rob	2 Western Ave, Waltham, MA 02876	Clash of titans	Matt

Checkpoint 1

Which of the following is Wrong?

A. An attribute of an entity can have more than one value

B. For A row of a relational table, an attribute can have more than one value

C. For A row of a relational table, an attribute can have exactly one value or a NULL value

Relational Databases in bigdata sets

- Pain points
  - Scaling
  - Sharding
  - Replication latencies
  - Performance (3NF)
    - Too many joins
  - Operational overhead

Non-Relational Databases – NoSQL

- While a SQL database is a defined, concrete concept, NoSQL is not.
- Dynamic schema for unstructured data
- data is stored in many ways:
  - column-oriented
  - document-oriented
  - Key Value store

Non-Relational Databases – NoSQL

Few good things…

- You can create documents without having to first define their structure
- Each document can have its own unique structure
- Do not require a defined schema structure
- Support wide range of modern programming languages and tools
- Primarily eventually-consistent by default

Popular Non-Relational Databases

- MongoDB by MongoDB Inc.
- DynamoDB by Amazon
- Cassandra from Apache foundation
- Coachbase from Apache foundation

Relational Vs Non-Relational
How to Decide?

- When it comes to choosing a database, one of the biggest decisions is picking SQL vs NoSQL

- **Relational (SQL) is a Strong choice for –**
    - Clear business rules and predefined structure
    - Applications that involve in transaction management
    - Link information from different tables to uniquely identify data sets
    - Maintain data integrity and relationships (Primary key, Foreign key)
    - Consistency
    - Support complex SQL to manage and maintain data

Relational Vs Non-Relational
How to Decide?

- Non-Relational (NoSQL) is a Strong choice for –
    - Any businesses that have rapid growth or databases with no clear schema definitions
    - Applications that cannot define a schema for database, if you find yourself de-normalizing data schemas, or if your schema continues to change
    - Very high performance
    - Eventually consistency

Recap

DBMS terms	Relational Terms
Table	Relation
Table Definition	Schema / Schema of relation
Records / Rows	Tuple(s)
Column / field	Attributes
Column value	Domain

Recap

- Cardinality of a relation is _____
- Degree of a relation is _____

DBMS terms	Relational Terms
Table	Relation
Table Definition	Schema / Schema of relation
Records / Rows	Tuple(s)
Column / field	Attributes
Column value	Domain

Recap

- **Properties of a relation?**
  - Has a name which is unique in its schema
  - Combination of row and column should have only one value
  - Every attribute in a relation should have unique names
  - Every row is unique

Recap

- ***Natural Key*** – *Identifier used to uniquely identify data*
- ***Super Key*** – *1 or more Column groups which can uniquely identify row. Eg: A tuple is always a super key*
- ***Candidate Key*** – *Least number of columns that can be used to uniquely identify a row (Subset of super key)*
- ***Primary Key*** – *Candidate key is eligible to become a primary key*

- ***Functional Dependency*** – Typically between PK and non Key columns – Help to reduce redundancy
  - ***Partial Dependency*** – *There is an attribute that depends on part of primary key*
  - ***Transitive Dependency*** – A → B, B → C, then **C** is transitively dependent on **A.**

- ***First Normal form*** – *No multivalued attribute, Entries in a column are of same type.*
- ***Second Normal form*** – *Should be in 1NF and No partial dependency*
- ***Third Normal form*** – *should be in 2NF and No Transitive dependency*

Recap

- **Multivalued attributes** - Not normalized yet
- **Removal of all Multivalued Attributes** gets you to First normal form (1NF)
- **Removal of Partial Dependencies** gets you to Second normal form (2NF)
- **Removal of Transitive Dependencies** gets you to Third normal form (3NF)

Recap

- When you normalize a relation *by* breaking it into two smaller relations, what must you do to maintain data integrity? (Select all that apply)

- A. Link the relations by a common field
- B. Remove any functional dependencies from both relations
- C. Assign both relations the same primary key field(s)
- D. Create a primary key(s) for the new relation

Recap

- When you normalize a relation by breaking it into two smaller relations, what must you do to maintain data integrity? (Select all that apply)

- A. **Link the relations by a common field**
- B. Remove any functional dependencies from both relations
- C. Assign both relations the same primary key field(s)
- D. **Create a primary key(s) for the new relation**

Recap

- Advantages of Normalization?

Recap

- Advantages of Normalization?

- More efficient data structure
- Avoid redundancy
- Easier to maintain data

- Any Disadvantages?

Introduction to SQL

- **Language used to access Data**
- **Key features**
  - ✓ Non Procedural language
  - ✓ Unified language
  - ✓ Common language for all Relational Databases

- DDL – Create objects like Tables, Views, Indexes
- DML – Modify data
- DCL – Controlling data and access

Datatypes

- **CHAR**
- **VARCHAR (VARCHAR2)**
- **NVARCHAR2**
- **NUMBER**
- **DATE**
- **BLOB**
- **CLOB**
- **RAW**

Syntax - SQL

```
SELECT      [COLUMN NAMES]
FROM        [TABLE NAMES]
WHERE       [JOIN CONDITIONS]
GROUP BY    [COLUMN NAMES]
ORDER BY    [COLUMN NAMES]
```

What’s in next week?

- **Relational Algebra**
- **SQL continues…**

Questions?