# Welcome to the CoGrammar

## Tutorial: Relational Databases

### The session will start shortly...

**Questions? Drop them in the chat. We'll have dedicated moderators answering questions.**

CoGrammar

# Software Engineering Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

# **Software Engineering Session Housekeeping** cont.

- For all **non-academic questions**, please submit a query: **www.hyperiondev.com/support**
- Report a **safeguarding** incident: **www.hyperiondev.com/safeguardreporting**
- We would love your **feedback** on lectures: **Feedback on Lectures**

# Skills Bootcamp
# 8-Week Progression Overview

## Fulfil 4 Criteria to Graduation

### ✅ Criterion 1: Initial Requirements

- ***Guided Learning Hours (GLH):*** Minimum of 15 hours
- ***Task Completion:*** First 4 tasks

**Due Date:** 24 March 2024

### ✅ Criterion 2: Mid-Course Progress

- ***Guided Learning Hours (GLH):*** Minimum of 60 hours
- ***Task Completion:*** *First* 13 tasks

**Due Date:** 28 April 2024

CoGrammar

# Skills Bootcamp Progression Overview

## ✅ Criterion 3: Course Progress

- ***Completion:*** All mandatory tasks, including Build Your Brand and resubmissions by study period end
- ***Interview Invitation:*** Within 4 weeks post-course
- ***Guided Learning Hours:*** Minimum of 112 hours by support end date (10.5 hours average, each week)

## ✅ Criterion 4: Demonstrating Employability

- ***Final Job or Apprenticeship Outcome:*** Document within 12 weeks post-graduation
- ***Relevance:*** Progression to employment or related opportunity

CoGrammar

# Learning Objectives & Outcomes

- Define Normalisation
- Normalise a table to 1NF, 2NF and 3NF
- Create tables with their relationships using SQLite.
- Connect your SQLite database to a Python program

CoGrammar

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar

# Relational Databases

April 2024

# Relational Databases

- A relational database is a database based on the relational model of data
- Relational Model
  - A relational model organizes data into one or more tables (or "relations") of columns and rows, with a unique key identifying each row.

CoGrammar

# Relational Databases

- Rows are also called records or tuples.
- Columns are also called attributes.
- Generally, each table/relation represents one "entity type" (such as customer or product).
- The rows represent instances of that type of entity and the columns represent values attributed to that instance.

CoGrammar

# Keys

- Each row in a table has its own unique key.
- Rows in a table can be linked to rows in other tables by adding a column for the unique key of the linked row
- Allows us to select or modify one and only one row in a table
- unique primary key (PK) for each row in a table
- Foreign and Primary keys

CoGrammar

# Relationships

- Relationships are a logical connection between different tables (entities), established based on interaction among these tables.
- These relationships can be modelled as an entity-relationship model.

CoGrammar

# Designing a Relational Database

1. Identify entities and their attributes.
   a. An entity is a real-world object or a concept. This can be a customer or an order.
   b. An attribute is a portion of an entity that's used to describe it, such as their name or the order ID.

2. Identify relationships between entities. An example would be customers place orders.
   a. These are typically identified and implemented using foreign keys

3. Database design should be built using normalization form.
   a. This is the process of organising data in different tables (Customers table and Order table)

CoGrammar

# Normalisation

- Process of organizing data in a database
- Creating tables and establishing relationships between those tables
- Eliminate redundancy and inconsistent dependency

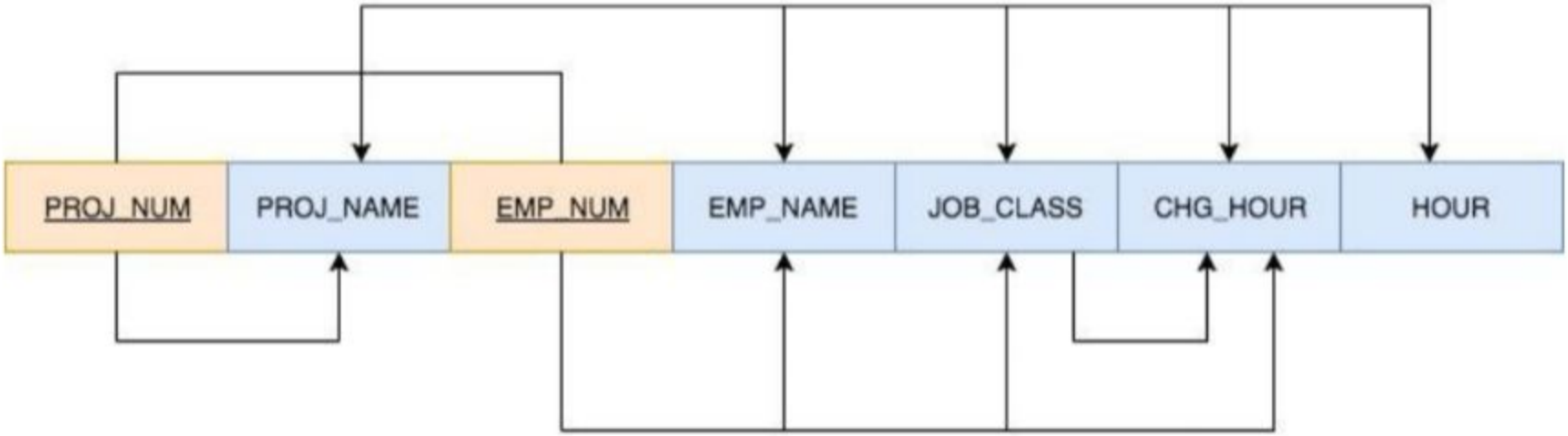
CoGrammar

# Normalisation

| PROJ_NUM | PROJ_NAME | EMP_NUM | EMP_NAME | JOB_CLASS | CHG_HOUR | HOURS |
|---|---|---|---|---|---|---|
| 15 | Evergreen | 103 | June Arbaugh | Elect. Engineer | $67.55 | 23 |
| 15 | Evergreen | 101 | John News | Database Designer | $82.00 | 19 |
| 15 | Evergreen | 105 | Alice Johnson | Database Designer | $82.00 | 35 |
| 15 | Evergreen | 106 | William Smithfield | Programmer | $26.66 | 12 |
| 15 | Evergreen | 102 | David Senior | System Analyst | $76.43 | 12 |
| 18 | Amberwave | 114 | Ann Jones | Applications Designer | $38.00 | 24 |
| 18 | Amberwave | 118 | James Frommer | General Support | $14.50 | 45 |
| 18 | Amberwave | 104 | Anne Remoras | System Analyst | $76.43 | 32 |
| 18 | Amberwave | 112 | Darlene Smithson | DSS Analyst | $36.30 | 44 |

CoGrammar

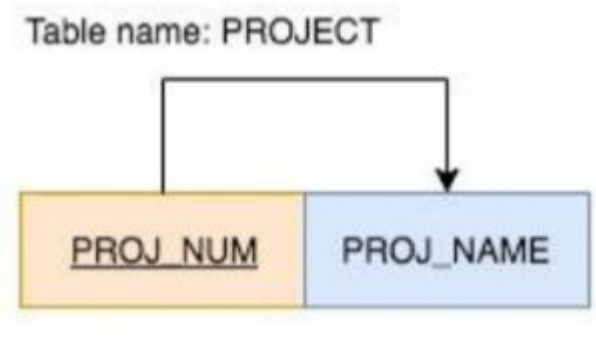
# 1NF

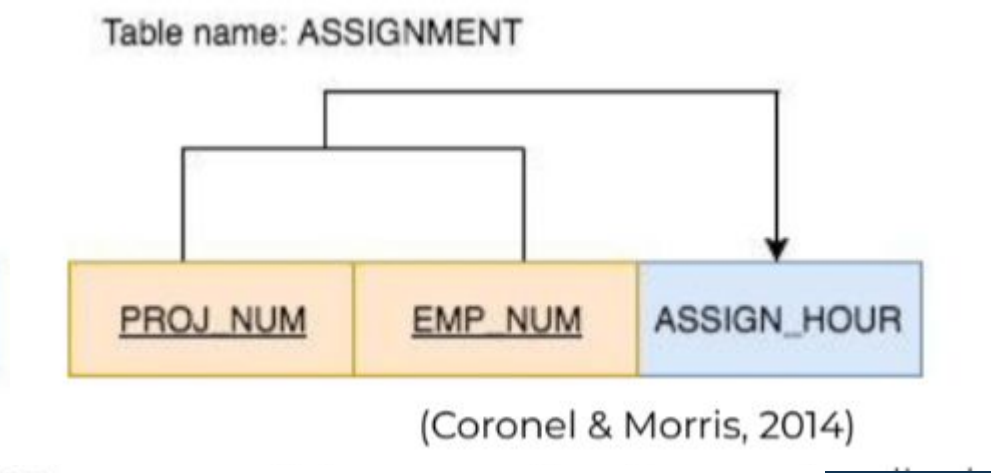



(Coronel & Morris, 2014, pg. 198)

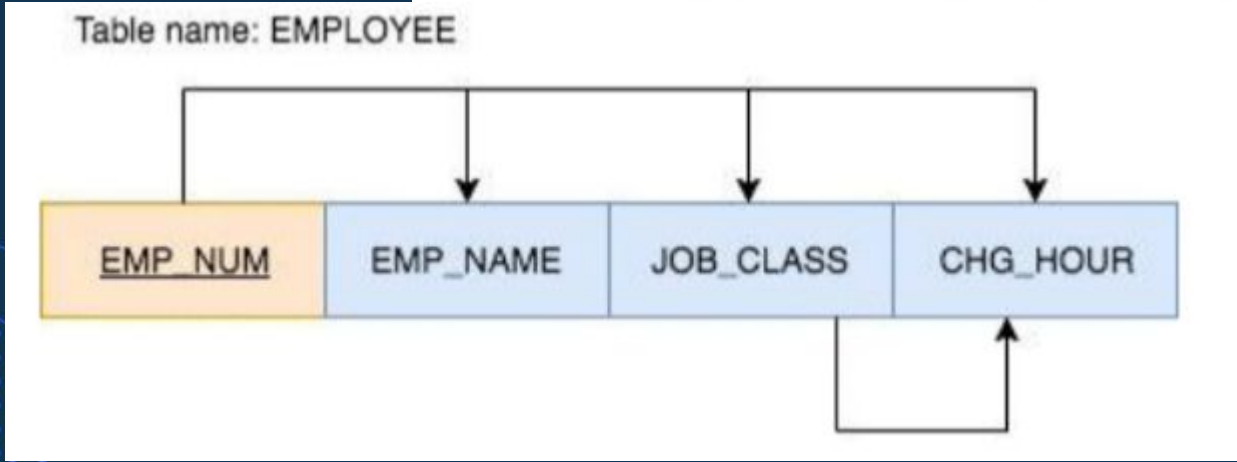
CoGrammar

# 2NF



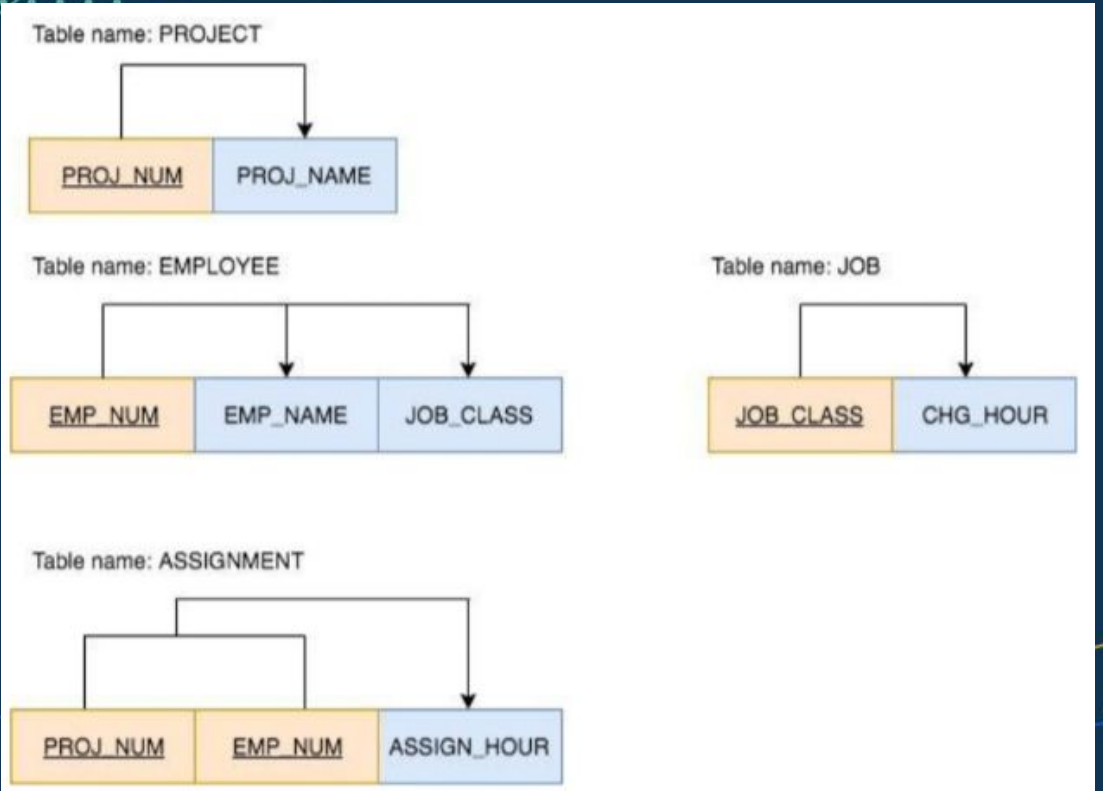




(Coronel & Morris, 2014)




CoGrammar

# 3NF




(Rob, Coronel, & Crockett, 2008, pg. 258)

CoGrammar

# DBMS

- Database management system (DBMS)
  - Collection of programs that manages the database structure and access to the data
  - Acts as intermediary between user and database
  - Hides internal complexity from the user

CoGrammar

# DBMS Advantages

- Better data sharing
  - End users have more efficient access to better managed data due to the DBMS managing the data and access to the data
- Improved data integration
  - The DBMS helps provide a clearer and more integrated view of the organisation's operations to the end-users

CoGrammar

# DBMS Advantages

- Minimised data inconsistency
  - Occurs when different versions of the same data appear in different places. Properly designed databases greatly reduces the probability of data inconsistency
- Improved data access
  - A query is a specific request for data manipulation sent to the DBMS. The DBMS makes it possible to produce quick answers to spur-of-the-moment queries

# DBMS Advantages

- Improved decision making
  - Better quality information (on which decisions are made) is generated due to better managed data and improved data access
- Increased end-user productivity
  - The availability of data and the ability to transform data in to usable information encourages end-users to make quicker and more informed decisions

CoGrammar

# Database Interaction: SQLite

- SQLite is a lightweight, self-contained SQL database engine that requires minimal setup and configuration. It is often used for smaller-scale projects or applications where simplicity and ease of use are prioritised.

CoGrammar

# SQLite: Key Features

- **Zero Configuration:** SQLite databases are self-contained, meaning they require no external server or setup. They are simply files that can be accessed by the application directly.
- **Single File:** The entire database is stored in a single file, making it easy to distribute and manage.
- **SQL Support:** SQLite supports standard SQL syntax for querying and manipulating data, making it compatible with existing SQL-based applications and tools.

# SQLite

- Native to Python (Yay! No pip installations!)
- Self-Contained
- Easy to port (Moving Database files)
- Serverless
- Doesn’t require client-server architecture. Works directly with files.
- Transactional
- Atomic, Consistent, Isolated and Durable (ACID).
- Ensures data integrity.

CoGrammar

# SQLite Syntax

```
import sqlite3

db = sqlite3.connect('data/student_db')
cursor = db.cursor()

cursor.execute('''
    CREATE TABLE student(id INTEGER PRIMARY KEY, name TEXT,
                grade INTEGER)
''')

db.commit()
```

CoGrammar

# Basic SQLite Syntax

```
cursor.execute('''INSERT INTO student(name, grade)
                VALUES(?,?)''', (name1,grade1))
db.commit()

students_ = [(name1,grade1),(name2,grade2),(name3,grade3)]
cursor.executemany(''' INSERT INTO student(name, grade) VALUES(?,?)''',
students_)
db.commit()
```

CoGrammar

# Let’s take a short break

CoGrammar

# Summary

- **Relational Databases:** Based on the relational model of data that organizes data into one or more tables (or "relations") of columns and rows, with a unique key identifying each row.
- **Normalisation:** Process of organizing data in a database.
- **DBMS:** Collection of programs that manages the database structure and access to the data
- **SQLite:** Lightweight, self-contained SQL database engine.

# Questions and Answers

CoGrammar

# Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar