

# SETTIMANA 1 – Database e AI

---

## Il Fondamento Necessario (4 ore su 60)

---

### LEZIONE 1 (3 ore)

---

#### Perché l'AI ha bisogno dei Database

---

##### Parte 1 – Il ruolo dei dati nell'AI (1,5h)

1. Opening provocatorio (15 min)

Demo concettuale (anche solo raccontata)

##### Scenario A – AI senza dati organizzati

“Dimmi quanti studenti hanno superato l'esame di SQL lo scorso anno”

Risposta vaga, inventata, incoerente → **allucinazione**

##### Scenario B – AI con database

Stessa domanda → query su database → risposta precisa

##### ☞ Domanda guida

“Cosa fa la differenza?”

##### RISPOSTA:

☞ DATI BEN ORGANIZZATI IN UN DATABASE

---

##### 2. Concetto chiave: l'AI NON è intelligente, è informata (15 min)

###### Senza database    Con database

---

Intuizioni vaghe    Risposte verificabili

---

Errori frequenti    Coerenza

---

Allucinazioni    Tracciabilità

##### ☞ Messaggio fondamentale per studenti

L'AI non “sa”, l'AI legge dati

##### 3. Training Data Pipeline (30 min)

### 3.1 Come viene addestrata un'AI (semplificato)

DATI → DATABASE → PREPROCESSING → TRAINING → MODELLO

#### Esempi concreti

##### GPT (testo)

- miliardi di frasi
- salvate in **database enormi**
- ogni riga = testo + metadati

##### Stable Diffusion (immagini)

- immagini
- descrizioni testuali
- tag
- autore
- licenza

☞ Senza database = caos

---

### 4. AI che usi ogni giorno = Database + AI (20 min)

Netflix (esempio intuitivo)

Cosa salva Netflix?

- utenti
- film
- visualizzazioni
- voti

L'AI non guarda film L'AI interroga database

---

Mini esempio concettuale (NO SQL ancora)

Utente	Film	Voto
Anna	Matrix	5
Anna	Inception	4
Luca	Matrix	5

☞ L'AI trova **pattern** grazie a **tabelle**

---

## 5. Tipi di dati nell'AI (20 min)

Tipo di dato	Esempio
Training data	testi, immagini storiche
Production data	click, sensori, log
User data	preferenze, chat
Model metadata	versione, accuratezza

### 🔗 Concetto DB

Ogni tipo di dato → **tabelle diverse**

---

## 6. Case Study – Midjourney (20 min)

Domanda chiave:

Come fa a trovare “un gatto steampunk” tra 100 milioni di immagini?

Risposta:

### ☞ Database + metadati

Esempio semplificato:

image_id	descrizione	tag
101	gatto con occhiali	gatto, steampunk
102	cane futuristico	cane, cyberpunk

AI = **motore di ricerca avanzato sui database**

---

## LEZIONE 2 (1 ora)

---

### Primo contatto con un Database

---

#### 1. Cos'è un database (10 min)

Un database è **una collezione organizzata di tabelle**

🔗 Come un **Excel**, ma:

- più sicuro
- più veloce
- interrogabile con SQL

## 2. Prima tabella: Dataset per AI (20 min)

Scenario: corso AI – studenti e esercizi

Tabella: `studenti`

```
CREATE TABLE studenti (
    id INTEGER,
    nome TEXT,
    corso TEXT
);
```

Inseriamo dati

```
INSERT INTO studenti VALUES (1, 'Anna', 'AI');
INSERT INTO studenti VALUES (2, 'Luca', 'AI');
INSERT INTO studenti VALUES (3, 'Marco', 'Web');
```

❖ Ogni riga = **dato utile per AI**

---

## 3. Prima query: leggere i dati (15 min)

```
SELECT * FROM studenti;
```

❖ L'AI non inventa, legge

---

## 4. Filtro: base di ogni AI (15 min)

```
SELECT *
FROM studenti
WHERE corso = 'AI';
```

❖ Messaggio chiave:

**Ogni AI fa filtri sui dati**

---

**COLLEGAMENTO ESPLICITO DATABASE ↔ AI  
(chiusura)**

---

## Senza SQL

- AI “intuisce”
- rischio errori

## Con SQL

- AI **verifica**
  - risposte affidabili
  - spiegabili
- 

# Output didattici della settimana 1

---

## Concetti chiave imparati

- l'AI **dipende dai database**
- i dati **devono essere strutturati**
- SQL è **linguaggio dell'AI**

✓ Nessuna complessità tecnica inutile ✓ Base solida per:

- JOIN
  - dataset AI
  - RAG
  - Data cleaning
  - analytics
- 

# ESERCIZI SETTIMANA 1

---

## DATASET DIDATTICI AI-READY

---

(piccoli, realistici, spiegabili)

### Dataset 1 – Studenti e Attività AI

#### ⌚ Obiettivo didattico

- capire cos'è un dataset
- abituarsi a “leggere dati” come fa un'AI

Tabella: **studenti**

```
CREATE TABLE studenti (
    id INTEGER,
    nome TEXT,
```

```
    corso TEXT,  
    livello TEXT  
);
```

## Dati

```
INSERT INTO studenti VALUES (1, 'Anna', 'AI', 'base');  
INSERT INTO studenti VALUES (2, 'Luca', 'AI', 'intermedio');  
INSERT INTO studenti VALUES (3, 'Marco', 'Web', 'base');  
INSERT INTO studenti VALUES (4, 'Sara', 'AI', 'base');
```

### 🔗 AI-ready perché

- attributi chiari
- categorizzazioni (corso, livello)
- dati interrogabili

---

## Dataset 2 – Dataset per raccomandazioni (stile Netflix)

### Tabella: contenuti

```
CREATE TABLE contenuti (  
    id INTEGER,  
    titolo TEXT,  
    categoria TEXT,  
    durata INTEGER  
);
```

```
INSERT INTO contenuti VALUES (1, 'Introduzione AI', 'AI', 30);  
INSERT INTO contenuti VALUES (2, 'SQL Base', 'Database', 45);  
INSERT INTO contenuti VALUES (3, 'Prompt Engineering', 'AI', 40);  
INSERT INTO contenuti VALUES (4, 'HTML Basics', 'Web', 25);
```

### 🔗 AI-ready perché

- simula dataset di raccomandazione
- attributi numerici + testuali

---

## Dataset 3 – Interazioni utente (fondamentale per AI)

### Tabella: visualizzazioni

```
CREATE TABLE visualizzazioni (
    id INTEGER,
    studente_id INTEGER,
    contenuto_id INTEGER,
    completato TEXT
);
```

```
INSERT INTO visualizzazioni VALUES (1, 1, 1, 'si');
INSERT INTO visualizzazioni VALUES (2, 1, 3, 'no');
INSERT INTO visualizzazioni VALUES (3, 2, 1, 'si');
INSERT INTO visualizzazioni VALUES (4, 4, 1, 'si');
```

### 💡 Concetto AI

L'AI impara **da ciò che gli utenti fanno**, non da ciò che dicono

## ESERCIZI SQL GUIDATAI

(progressivi – “ragiona come un'AI”)

### ESERCIZI LIVELLO 1 – LEGGERE I DATI

#### Esercizio 1 – Guarda tutto il dataset

☞ Come fa un'AI a “conoscere” i dati?

```
SELECT * FROM studenti;
```

### 🧠 Ragionamento AI

Prima osserva, poi decide

#### Esercizio 2 – Elenco dei contenuti disponibili

```
SELECT * FROM contenuti;
```

### ESERCIZI LIVELLO 2 – FILTRARE (CUORE DELL'AI)

#### Esercizio 3 – Studenti del corso AI

```
SELECT *
FROM studenti
WHERE corso = 'AI';
```

### 🔗 Collegamento AI

Questo è esattamente ciò che fa un sistema di raccomandazione

## Esercizio 4 – Contenuti di AI

```
SELECT titolo
FROM contenuti
WHERE categoria = 'AI';
```

## ESERCIZI LIVELLO 3 – DECISIONI SUI DATI

### Esercizio 5 – Studenti AI livello base

```
SELECT nome
FROM studenti
WHERE corso = 'AI'
AND livello = 'base';
```

### ⌚ AI mindset

Segmentazione degli utenti

### Esercizio 6 – Contenuti brevi ( $\leq$ 30 minuti)

```
SELECT titolo, durata
FROM contenuti
WHERE durata <= 30;
```

### 🔗 Tipico uso AI

- suggerimenti “veloci”
- micro-learning

## ESERCIZI LIVELLO 4 – INTERAZIONI (DATI COMPORTAMENTALI)

### Esercizio 7 – Contenuti completati

```
SELECT *
FROM visualizzazioni
WHERE completato = 'si';
```

### ❖ Messaggio chiave

L'AI si fida più dei comportamenti che delle parole

---

### Esercizio 8 – Chi NON ha completato un contenuto

```
SELECT *
FROM visualizzazioni
WHERE completato = 'no';
```

☞ base per:

- tutoring AI
  - adaptive learning
- 

## ESERCIZI DI RAGIONAMENTO (NO SQL)

---

### Domanda 1

Quali dati servirebbero per suggerire un nuovo contenuto a Sara?

✓ corso ✓ contenuti visti ✓ completamento

---

### Domanda 2

Perché un'AI senza la tabella **visualizzazioni** sarebbe "cieca"?

---

## OUTPUT DIDATTICI (fine settimana 1)

---

Cosa sai fare dopo le prime 4 ore

✓ dataset AI ✓ leggere e filtrare dati ✓ ragionare come un'AI ✓ SQL come strumento decisionale

---

## SETTIMANA 2

---

JOIN = AI Reasoning

*Collegare dati = pensare*

## OBIETTIVO DELLA SETTIMANA

Al termine lo studente saprà:

- ✓ capire perché JOIN = intelligenza ✓ collegare tabelle come fa un'AI ✓ scrivere JOIN semplici ✓ interpretare risultati come decisioni AI

### **Messaggio chiave**

Un'AI non è intelligente perché calcola È intelligente perché **mette in relazione i dati**

## LEZIONE 1 (3 ore)

### Perché le AI usano JOIN

#### 1. Opening provocatorio (15 min)

Domanda:

"Quali contenuti AI consigliare a Sara?"

Problema:

- studenti → una tabella
- contenuti → un'altra tabella
- visualizzazioni → un'altra

 **Senza JOIN → risposta impossibile**

#### 2. Concetto chiave: JOIN = connessioni (20 min)

Senza JOIN (dati isolati)

Tabella	Informazione
studenti	chi
contenuti	cosa
visualizzazioni	comportamento

Con JOIN (ragionamento)

Chi → cosa → come → perché

 **AI = sistema che connette informazioni**

---

# DATASET SETTIMANA 2 (ripasso + estensione)

---

Tabella: studenti

```
CREATE TABLE studenti (
    id INTEGER,
    nome TEXT,
    corso TEXT,
    livello TEXT
);
```

Tabella: contenuti

```
CREATE TABLE contenuti (
    id INTEGER,
    titolo TEXT,
    categoria TEXT,
    durata INTEGER
);
```

Tabella: visualizzazioni

```
CREATE TABLE visualizzazioni (
    id INTEGER,
    studente_id INTEGER,
    contenuto_id INTEGER,
    completato TEXT
);
```

☞ Questa struttura è tipica di:

- e-learning AI
- Netflix
- Spotify
- ChatGPT logs

---

## 3. JOIN spiegato “come a un umano” (30 min)

---

## Metafora

JOIN è come dire: "Prendi queste informazioni e collegale a queste altre"

### Sintassi base (INNER JOIN)

```
SELECT ...
FROM tabella1
JOIN tabella2 ON condizione;
```

☞ **ON = regola di collegamento**

## 4. Primo JOIN – Chi ha visto cosa (45 min)

### Obiettivo AI

Collegare **studenti** e **visualizzazioni**

```
SELECT studenti.nome, visualizzazioni.contenuto_id
FROM studenti
JOIN visualizzazioni
ON studenti.id = visualizzazioni.studente_id;
```

### 🧠 Ragionamento AI

Chi → cosa ha fatto

## 5. JOIN con tre tavole – Raccomandazione (45 min)

### Domanda AI reale

Che contenuti ha visto ogni studente?

```
SELECT
    studenti.nome,
    contenuti.titolo,
    visualizzazioni.completato
FROM studenti
JOIN visualizzazioni
    ON studenti.id = visualizzazioni.studente_id
```

```
JOIN contenuti  
    ON contenuti.id = visualizzazioni.contenuto_id;
```

---

### 🔗 Questo è AI reasoning

---

## LEZIONE 2 (3 ore)

---

### JOIN per decisioni intelligenti

---

#### 6. Filtro + JOIN = decisione (30 min)

Studenti AI che NON hanno completato contenuti AI

```
SELECT studenti.nome, contenuti.titolo  
FROM studenti  
JOIN visualizzazioni  
    ON studenti.id = visualizzazioni.studente_id  
JOIN contenuti  
    ON contenuti.id = visualizzazioni.contenuto_id  
WHERE studenti.corso = 'AI'  
AND visualizzazioni.completato = 'no';
```

---

### 🔗 Use case

- tutoring AI
  - supporto personalizzato
- 

#### 7. JOIN + aggregazioni (30 min)

Contenuti più completati

```
SELECT contenuti.titolo, COUNT(*) AS completamenti  
FROM visualizzazioni  
JOIN contenuti  
    ON contenuti.id = visualizzazioni.contenuto_id  
WHERE visualizzazioni.completato = 'si'  
GROUP BY contenuti.titolo;
```

---

### 🔗 Analytics AI

---

#### 8. JOIN = base del RAG (20 min)

## Concetto

SQL	RAG
JOIN	recupero contesto
WHERE	filtro rilevanza
GROUP BY	sintesi
LIMIT	top-k

☞ **RAG = JOIN + embeddings**

---

## 9. LEFT JOIN – ciò che manca (20 min)

Chi NON ha visto nulla?

```
SELECT studenti.nome  
FROM studenti  
LEFT JOIN visualizzazioni  
ON studenti.id = visualizzazioni.studente_id  
WHERE visualizzazioni.id IS NULL;
```

☞ **AI insight**

Anche l'assenza di dati è informazione

---

## LAB SETTIMANA 2 (90 min)

---

### LAB 1 – Ricostruire il profilo studente

"Cosa sappiamo di Anna?"

- nome
- corso
- contenuti visti
- completamenti

JOIN libero guidato

---

### LAB 2 – Simulare una raccomandazione AI

Domanda:

"Consiglia un contenuto AI a uno studente AI livello base"

Passaggi:

1. individua studente
2. vedi cosa ha già visto
3. suggerisci ciò che manca

☞ **Nessuna AI “magica”, solo dati collegati**

---

## ESERCIZI DI RIFLESSIONE (NO SQL)

---

1. Perché un'AI senza JOIN è “stupida”?
  2. Quali JOIN servono a ChatGPT per rispondere meglio?
  3. JOIN o Vector DB? Quando uno, quando l'altro?
- 

## OUTPUT DIDATTICI SETTIMANA 2

---

Gli studenti: ✓ sanno usare JOIN ✓ comprendono l'AI come **sistema di relazioni** ✓ leggono query come **ragionamenti** ✓ sono pronti per RAG e analytics avanzati

---

## SETTIMANA 3

---

Aggregazioni = Insight AI

**Durata suggerita: 6 ore (2 × 3h)**

---

### OBIETTIVO DELLA SETTIMANA

Al termine lo studente saprà:

✓ usare COUNT, SUM, AVG, MIN, MAX ✓ combinare JOIN + GROUP BY ✓ interpretare risultati come **decisioni AI** ✓ capire perché le AI usano statistiche, non singoli dati

☞ **Messaggio chiave**

L'AI non guarda i singoli record L'AI guarda **pattern aggregati**

---

## LEZIONE 1 (3 ore)

---

Dati → Pattern → Decisioni

---

1. Opening provocatorio (15 min)

Domanda:

“Questo corso AI funziona?”

Singolo studente → **non basta** Media, frequenze, trend → **insight**

☞ **Aggregazioni = intelligenza**

---

## 2. Ripasso concettuale (15 min)

Operazione	Significato
JOIN	collega dati
WHERE	filtra
GROUP BY	raggruppa
AGGREGATE	riassume

---

## FUNZIONI AGGREGATE (CORE)

---

## 3. COUNT – contare eventi (30 min)

Domanda AI

Quanti studenti sono iscritti ai corsi?

```
SELECT corso, COUNT(*) AS numero_studenti  
FROM studenti  
GROUP BY corso;
```

☞ **Insight**

L'AI capisce dove investire risorse

---

## 4. COUNT + JOIN – engagement (30 min)

Quanti studenti hanno completato contenuti?

```
SELECT studenti.nome, COUNT(*) AS completamenti  
FROM studenti  
JOIN visualizzazioni  
    ON studenti.id = visualizzazioni.studente_id  
WHERE visualizzazioni.completato = 'si'  
GROUP BY studenti.nome;
```

### Use case AI

- ranking utenti
  - detection inattività
- 

## 5. AVG – performance media (30 min)

Durata media dei contenuti visti

```
SELECT studenti.nome, AVG(contenuti.durata) AS durata_media
FROM studenti
JOIN visualizzazioni
    ON studenti.id = visualizzazioni.studente_id
JOIN contenuti
    ON contenuti.id = visualizzazioni.contenuto_id
GROUP BY studenti.nome;
```

### Insight

contenuti troppo lunghi? troppo corti?

---

## LEZIONE 2 (3 ore)

Aggregazioni = cervello dell'AI

---

## 6. GROUP BY multiplo – segmentazione (30 min)

Studenti per corso e livello

```
SELECT corso, livello, COUNT(*) AS totale
FROM studenti
GROUP BY corso, livello;
```

### AI reale

- segmentazione utenti
  - personalizzazione
- 

## 7. Contenuti più efficaci (30 min)

Quali contenuti vengono completati di più?

---

```
SELECT contenuti.titolo, COUNT(*) AS completamenti
FROM contenuti
JOIN visualizzazioni
    ON contenuti.id = visualizzazioni.contenuto_id
WHERE visualizzazioni.completato = 'si'
GROUP BY contenuti.titolo
ORDER BY completamenti DESC;
```

## 🔗 Recommender systems

### 8. HAVING – filtro sugli insight (20 min)

Contenuti con almeno 2 completamenti

```
SELECT contenuti.titolo, COUNT(*) AS completamenti
FROM contenuti
JOIN visualizzazioni
    ON contenuti.id = visualizzazioni.contenuto_id
WHERE visualizzazioni.completato = 'si'
GROUP BY contenuti.titolo
HAVING COUNT(*) >= 2;
```

## 🔗 Insight selettivo

non tutti i pattern sono rilevanti

### 9. MIN / MAX – limiti e anomalie (20 min)

Contenuti più lunghi e più brevi

```
SELECT
    MIN(durata) AS durata_min,
    MAX(durata) AS durata_max
FROM contenuti;
```

## 🔗 AI monitoring

- outlier
- anomalie

## LAB SETTIMANA 3 (90 min)

## LAB 1 – Cruscotto AI (45 min)

Gli studenti devono rispondere a:

1. Quanti studenti per corso?
2. Qual è il contenuto più completato?
3. Chi è lo studente più attivo?
4. Qual è la durata media dei contenuti visti?

☞ solo SQL aggregato

---

## LAB 2 – Decisioni AI simulate (45 min)

Scenario:

Sei un'AI tutor. Devi decidere:

- quali contenuti migliorare
- quali studenti aiutare
- dove investire

Gli studenti giustificano le decisioni **con query**

☞ SQL come spiegabilità AI

---

## ESERCIZI DI RAGIONAMENTO (NO SQL)

1. Perché un'AI senza aggregazioni "vede tutto piatto"?
  2. Differenza tra dato e insight?
  3. Aggregazioni vs Vector DB: cosa fanno di diverso?
- 

## OUTPUT DIDATTICI SETTIMANA 3

Gli studenti: ✓ capiscono il concetto di **pattern** ✓ sanno sintetizzare dati ✓ leggono query come **decisioni**  
✓ sono pronti per analytics AI reali