UNIVERSITÀ DELLA CALABRIA

Facoltà di SS.MM.FF.NN.

Corso di Laurea in Informatica

A.A. 2019/2020

PROGETTO PER IL CORSO DI

BASI DI DATI RELAZIONALI

DOCENTE: PROF. P. RULLO

LABORATORIO: ING. G. LABOCCETTA,
DOTT.SSA D. ANGILICA

SISTEMA INFORMATIVO

PER LA GESTIONE DI

UN ISTITUTO COMPRENSIVO.

GRUPPO 39
<201018, Avolio Daniele>
<200850, Bilotta Simone>
<200771, Fazio Alessandro>

1. TEMATICA PROGETTUALE

La progettazione del sistema informatico in esame riguarda la gestione di un istituto comprensivo.

2. RACCOLTA E ANALISI DEI REQUISITI

2.1. Raccolta dei requisiti

	REQUISITI RICHIESTI
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17	L'istituto comprensivo StudioOnline vuole riorganizzare il proprio database a seguito di alcuni aggiornamenti nella normativa. Il complesso scolastico include tre gradi: scuola dell'infanzia, scuola primaria e scuola secondaria inferiore. Ciascuna scuola è distribuita in diversi plessi e ciascun plesso ospita anche più di un grado scolastico. Le iscrizioni avvengono nel mese di gennaio: i genitori iscrivono i propri figli e di ognuno di loro viene registrata l'anagrafica. Ogni studente, in un anno scolastico (01/09-19/06), è iscritto ad uno dei 3 gradi dell'istituto ma si vuole memorizzare per ciascun ragazzo lo storico delle iscrizioni. All'atto dell'iscrizione, ogni studente è associato ad una classe. I plessi sono identificati da un indirizzo e sono composti da delle aule. Di ogni aula si conoscono i metri quadri, hanno un numero massimo di studenti ospitabili e la fascia di età degli studenti ospitabili. Ogni classe, che ogni anno può cambiare aula, è identificata da un numero (1-4 per l'infanzia, 1-5 per la primaria e 1-3 per le medie) e da una lettera dell'alfabeto (aule di gradi diversi possono avere la stessa coppia). Di ciascun insegnante si conosce, ogni anno, il numero di ore lavorative settimanali e le classi a cui è assegnato.

2.1.1. Eliminazione delle ambiguità

LINEE	TERMINE	SINONIMI	MOTIVAZIONE CORREZIONE
3	Complesso scolastico	Istituto comprensivo	Semplificazione del termine
3	Scuola primaria	Scuola elementare	Semplificazione del termine
3	Scuola dell'infanzia	Asilo	Semplificazione del termine
4	Scuola secondaria inferiore	Scuola media	Semplificazione del termine
4,5 , 11	Plesso	Luogo fisico / Edificio scolastico con più gradi all'interno	Semplificazione del termine per una migliore comprensione
13	Fascia d'età	Età minima ed età massima	Semplificazione del termine
14	Classe	Insieme di studenti	Semplificazione del termine
14	Aula	Luogo fisico dove vengono svolte le lezioni	Semplificazione del termine

2.1.2. Ristrutturazione dei requisiti richiesti

	REQUISITI RICHIESTI
1	L'istituto comprensivo StudioOnline vuole riorganizzare il proprio database a seguito
2	di alcuni aggiornamenti nella normativa.
3	L'istituto comprensivo include tre gradi: asilo, scuola elementare e scuola media.
4	Ciascuna scuola è distribuita in un edificio scolastico e ciascun edificio scolastico
5	ospita anche più di un grado scolastico.
6	Le iscrizioni avvengono nel mese di gennaio: i genitori iscrivono i propri figli e di
7	ognuno di loro viene registrata l'anagrafica.
8	Ogni studente, in un anno scolastico (01/09-19/06), è iscritto ad uno dei 3 gradi
9	dell'istituto ma si vuole memorizzare per ciascun ragazzo lo storico delle iscrizioni.
10	All'atto dell'iscrizione, ogni studente è associato ad una classe.
11	Gli edifici scolastici sono identificati da un indirizzo e sono composti da delle aule. Di
12	ogni aula si conoscono i metri quadri, hanno un numero massimo di studenti ospitabili
13	e la loro età minima ed età massima.
14	Ogni classe, che ogni anno può cambiare aula, è identificata da un numero (1-4 per
15	l'infanzia, 1-5 per la primaria e 1-3 per le medie) e da una lettera dell'alfabeto (aule di
16	gradi diversi possono avere la stessa coppia).
17	Di ciascun insegnante si conosce, ogni anno, il numero di ore lavorative settimanali e
18	le classi a cui è assegnato.

2.1.3. Raffinamento delle Specifiche e Individuazione dei Concetti di Base

	Frasi di carattere generale
1	L'istituto comprensivo include tre gradi: asilo, scuola elementare e scuola media.
2	Ciascuna scuola è distribuita in un edificio scolastico e ciascun edificio scolastico
3	ospita anche più di un grado scolastico.

	Frasi relative agli edifice, alle classi e aule
4	Gli edifici scolastici sono identificati da un indirizzo e sono composti da delle aule.
5	Ogni classe, che ogni anno può cambiare aula, è identificata da un numero (1-4 per
6	l'infanzia, 1-5 per la primaria e 1-3 per le medie) e da una lettera dell'alfabeto (aule di
7	gradi diversi possono avere la stessa coppia).
8	Di ogni aula si conoscono i metri quadri, hanno un numero massimo di studenti
9	ospitabili e la loro età minima ed età massima.

	Frasi relative ai professori
11	Di ciascun insegnante si conosce, ogni anno, il numero di ore lavorative settimanali e
12	le classi a cui è assegnato.

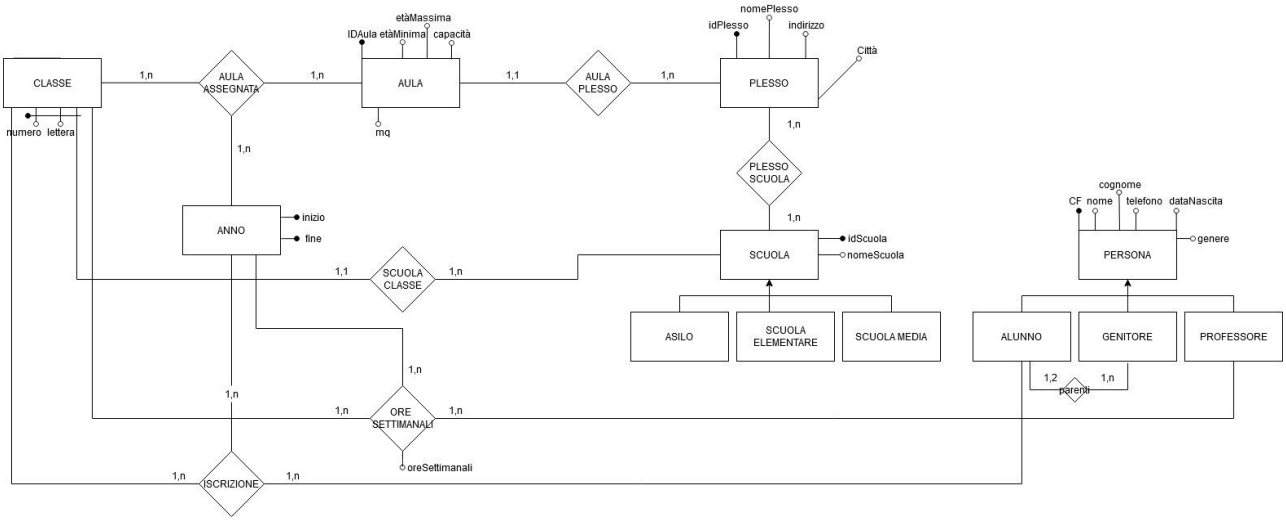
	Frasi relative agli studenti
13	Ogni studente, in un anno scolastico (01/09-19/06), è iscritto ad uno dei 3 gradi
14	dell'istituto ma si vuole memorizzare per ciascun ragazzo lo storico delle iscrizioni.
15	All'atto dell'iscrizione, ogni studente è associato ad una classe.

2.2. Specifica delle Operazioni sui dati previste

1. Inserire una nuova iscrizione.
2. Trovare gli studenti iscritti alla scuola secondaria inferiore nell'anno 2020.
3. Trovare gli studenti che hanno frequentato almeno una classe per ogni grado dell'istituto.
4. Trovare i plessi con il numero di studenti ospitabili più alto.
5. Trovare, per ogni anno, i gradi con il maggior numero di iscritti.
6. Controllare che ad un'aula non vengano assegnati più iscritti di quanti ne possa ospitare.
7. Alla scadenza delle iscrizioni annuali (1 febbraio) si memorizzi il numero di nuovi iscritti per grado e classe (1-4 infanzia, 1-5 primaria e 1-3 medie)

3. PROGETTAZIONE CONCETTUALE

3.1. Schemi E-R



3.2. Documentazione dello schema E-R

3.2.1. Dizionario dei dati

Entità	descrizione	attributi	identificatore
Persona	Rappresenta la persona che interagisce con l'istituto	CF, nome, cognome, telefono, dataNascita, genere	CF
Alunno	Rappresenta lo studente, e fa parte della generalizzazione di Persona	CF, nome, cognome, telefono, dataNascita, genere	CF
Scuola	Rappresenta il luogo giuridico di istruzione	idScuola, nomeScuola	idScuola
Plesso	Rappresenta il luogo fisico / edificio	idPlesso, nomePlesso, indirizzo	idPlesso
Aula	Rappresenta il luogo fisico dove vengono svolte le lezioni	idAula, etaMinima, etaMassima, capacità mq	idAula
Classe	Rappresenta l'insieme degli studenti	Numero, lettera,	numero, lettera
Anno	Identifica l'anno corrente	Inizio, fine	Inizio, fine

3.2.2. Descrizione entità

Persona			
Descrizione	Rappresenta la persona		
Nome Attributo	Tipo di dato	Vincolo	Descrizione
CF	Alfanumerico	16 caratteri	Identifica la persona
Nome	Alfanumerico		Specifica il nome della persona
Cognome	Alfanumerico		Specifica il cognome della persona
DataNascita	Data		Specifica la data di nascita della persona
Telefono	Numerico	10 caratteri	Specifica il numero telefonico della persona
Genere	Alfanumerico		Specifica il sesso della persona

Aula			
Descrizione	Rappresenta fisicamente le caratteristiche di un'aula.		
Nome Attributo	Tipo di dato	Vincolo	Descrizione
idAula	Numerico		Identifica l'aula
etaMinima	Numerico		Specifica l’età minima
etaMassima	Numerico		Specifica l’età massima
Mq	Numerico		Specifica i metri quadri dell’aula
Capacità	Numerico		Specifica la capienza dell’aula

Plesso			
Descrizione	Specifica le caratteristiche del plesso		
Nome Attributo	Tipo di dato	Vincolo	Descrizione
idPlesso	Numerici		Identifica univocamente il plesso
NomePlesso	Alfanumerico		Identifica il nome del plesso
Indirizzo	Alfanumerico		Identifica l'indirizzo del plesso
Città	Alfanumerico		Identifica la città del plesso

3.2.3. Descrizione relazioni

Parenti		
Descrizione	Indica il legame di parentela tra alunno e genitore	
Entità Coinvolte		
Entità	**Cardinalità**	
Alunno	(1,2)	
Genitore	(1,N)	
Attributi		
Nome	**Tipo di dati**	**Descrizione**

Iscrizione		
Descrizione	Rappresenta l'iscrizione di un alunno in una determinata classe in un dato anno	
Entità Coinvolte		
Entità	**Cardinalità**	
Anno	(1,N)	
Alunno	(1,N)	
Classe	(1,N)	
Attributi		
Nome	**Tipo di dati**	**Descrizione**

Aula Assegnata		
Descrizione	Rappresenta l'aula che viene assegnata ad una classe in un dato anno	
Entità Coinvolte		
Entità	**Cardinalità**	
Anno	(1,N)	
Aula	(1,N)	
Classe	(1,N)	
Attributi		
Nome	**Tipo di dati**	**Descrizione**

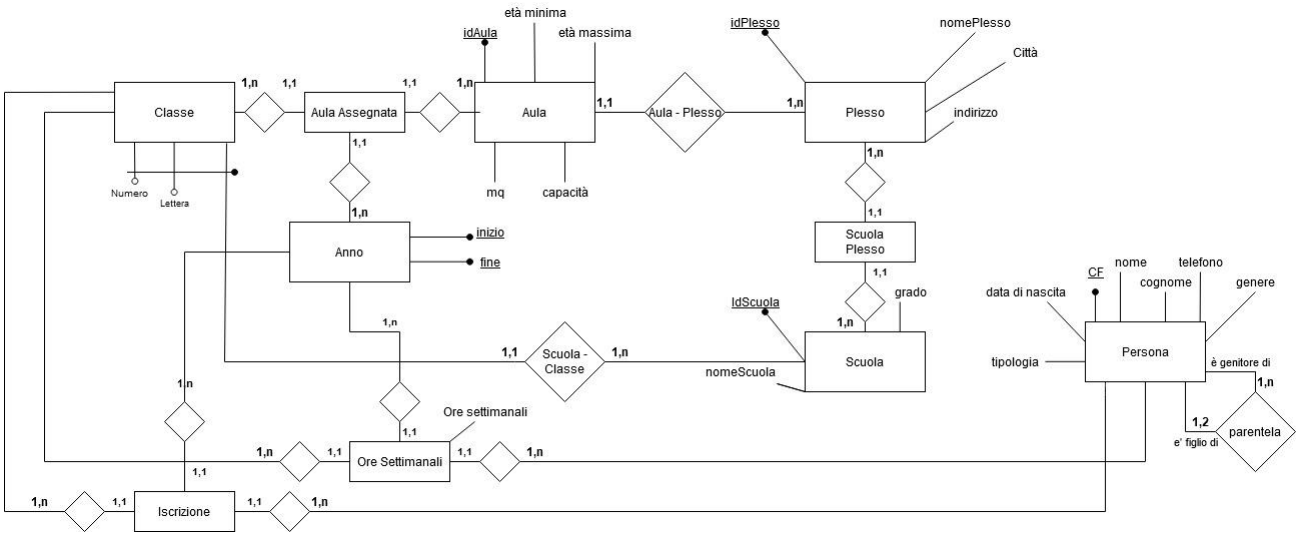
3.2.4. Vincoli non espressi dallo schema E/R

Regole di Vincolo
Bisogna rispettare la capienza massima di una classe
Ci si può iscrivere solo nel mese di Gennaio
L'inizio dell'anno dev'essere sempre inferiore alla fine dell'anno
Un'aula deve contenere solo iscritti rientranti in una determinata fascia d'età

4. PROGETTAZIONE LOGICA

Il modello logico utilizzato in questo progetto didattico è il Modello **Relazionale**.

4.1. Ristrutturazione dello Schema E-R



4.2. Documentazione dello Schema Logico

4.2.1. Descrizione entità

Persona			
Descrizione	Rappresenta la persona		
Nome Attributo	Tipo di dato	Vincolo	Descrizione
CF	CHAR(16)	16 caratteri	Identifica la persona
Nome	VARCHAR(50)		Specifica il nome della persona
Cognome	VARCHAR(50)		Specifica il cognome della persona
DataNascita	Date		Specifica la data di nascita della persona
Telefono	CHAR(10)	10 caratteri	Specifica il numero telefonico della persona
Genere	SET('m', 'f')		Specifica il sesso della persona

Aula			
Descrizione	Rappresenta fisicamente le caratteristiche di un'aula.		
Nome Attributo	Tipo di dato	Vincolo	Descrizione
idAula	INTEGER		Identifica l'aula
etaMinima	INTEGER		Specifica l’età minima
etaMassima	INTEGER		Specifica l’età massima
Mq	INTEGER		Specifica i metri quadri dell’aula
Capacità	INTEGER		Specifica la capienza dell’aula

Plesso			
Descrizione	Specifica le caratteristiche del plesso		
Nome Attributo	Tipo di dato	Vincolo	Descrizione
idPlesso	INTEGER		Identifica univocamente il plesso
NomePlesso	VARCHAR(50)		Identifica il nome del plesso
Indirizzo	VARCHAR(100)		Identifica l’indirizzo del plesso
Città	VARCHAR(50)		Identifica la città del plesso

4.2.2. Descrizione Relazioni

Aula - Plesso	
Descrizione	Associa le aule ad un plesso
Entità Coinvolte	
Entità	**Cardinalità**
Aula	(1,1)
Plesso	(1,N)

4.2.3. Traduzione dello Schema E-R

4.2.3.1. Traduzione Entità

Scuola (IdScuola, nomeScuola, grado)

Plesso (IdPlesso, nomePlesso, indirizzo, città)

Aula (IdAula, mq, etàMinima, etàMassima, capacità, plessoFK*)

Esiste un vincolo di integrità referenziale tra l'attributo plessoFK e la relazione Plesso

Classe (Numero, lettera, scuolaFK*)

Esiste un vincolo di integrità referenziale tra l'attributo scuolaFK e la relazione Scuola

Anno (Inizio, Fine)

Persona (CF, nome, cognome, telefono, genere, dataNascita, tipologia)

AulaAssegnata ((numeroFK, letteraFK, scuolaFK)* , (inizioFK, fineFK)* , idAulaFK*)

Esiste un vincolo di integrità referenziale tra numeroFK, letteraFK, scuolaFK e la relazione Classe, tra inizioFK, fineFK e la relazione Anno, tra idAulaFK e la relazione Aula

Iscrizione (CFFK*, (numeroFK, letteraFK, scuolaFK)*, (inizioFK, fineFK)*)

Esiste un vincolo di integrità referenziale tra CFFK e la relazione Persona, tra numeroFK, letteraFK, scuolaFK e la relazione Classe, tra inizioFK, fineFK e la relazione Anno.

OreSettimanali(CFFK*, (numeroFK, letteraFK, scuolaFK)*, (inizioFK, fineFK)*, oreSettimanali)

Esiste un vincolo di integrità referenziale tra CFFK e la relazione Persona, tra numeroFK, letteraFK, scuolaFK e la relazione Classe, tra inizioFK, fineFK e la relazione Anno.

ScuolaPlesso (scuolaFK*, plessoFK*)

Esiste un vincolo di integrità referenziale tra scuolaFK e la relazione Scuola, tra plessoFK e la relazione Plesso.

4.2.3.2. Traduzione Relazioni

Parentela (genitore*, figlio*)

Esiste un vincolo di integrità referenziale tra genitore e la relzione Persona, tra figlio e la relazione Persona.

5. PROGETTAZIONE FISICA

5.1. Definizione dello schema della base di dati

```
CREATE TABLE `anno` (
  `inizio` int NOT NULL,
  `fine` int NOT NULL,
  PRIMARY KEY (`inizio`,`fine`);

CREATE TABLE `aula` (
  `idAula` int NOT NULL AUTO_INCREMENT,
  `mq` int NOT NULL,
  `etàMinima` int NOT NULL,
  `etàMassima` int DEFAULT NULL,
  `capacità` int NOT NULL,
  `plessoFK` int NOT NULL,
  PRIMARY KEY (`idAula`),
  KEY `fk_Aula_Plesso1_idx` (`plessoFK`),
  CONSTRAINT `fk_Aula_Plesso1` FOREIGN KEY (`plessoFK`) REFERENCES `plesso`
(`idPlesso`);

CREATE TABLE `aulaassegnata` (
  `numeroFK` int NOT NULL,
  `letteraFK` char(1) NOT NULL,
  `scuolaFK` int NOT NULL,
  `inizioFK` int NOT NULL,
  `fineFK` int NOT NULL,
  `idAulaFK` int NOT NULL,
  PRIMARY KEY (`numeroFK`,`letteraFK`,`scuolaFK`,`inizioFK`,`fineFK`),
  KEY `fk_AulaAssegnata_Anno1_idx` (`inizioFK`,`fineFK`),
  KEY `fk_AulaAssegnata_Classe1_idx` (`numeroFK`,`letteraFK`,`scuolaFK`),
  KEY `fk_AulaAssegnata_Aula1` (`idAulaFK`),
  CONSTRAINT `fk_AulaAssegnata_Anno1` FOREIGN KEY (`inizioFK`, `fineFK`)
REFERENCES `anno` (`inizio`, `fine`),
  CONSTRAINT `fk_AulaAssegnata_Aula1` FOREIGN KEY (`idAulaFK`) REFERENCES
`aula` (`idAula`),
  CONSTRAINT `fk_AulaAssegnata_Classe1` FOREIGN KEY (`numeroFK`, `letteraFK`,
`scuolaFK`) REFERENCES `classe` (`numero`, `lettera`, `scuolaFK`);

CREATE TABLE `classe` (
  `numero` int NOT NULL,
  `lettera` char(1) NOT NULL,
  `scuolaFK` int NOT NULL,
  PRIMARY KEY (`numero`,`lettera`,`scuolaFK`),
  KEY `fk_Classe_Scuola1_idx` (`scuolaFK`),
  CONSTRAINT `fk_Classe_Scuola1` FOREIGN KEY (`scuolaFK`) REFERENCES
`scuola` (`idScuola`);
```

```
CREATE TABLE `iscrizione` (
  `CFFK` char(16) NOT NULL,
  `numeroFK` int NOT NULL,
  `letteraFK` char(1) NOT NULL,
  `scuolaFK` int NOT NULL,
  `inizioFK` int NOT NULL,
  `fineFK` int NOT NULL,
  PRIMARY KEY (`CFFK`,`numeroFK`,`letteraFK`,`fineFK`,`inizioFK`,`scuolaFK`),
  KEY `fk_Iscrizione_Classe1_idx` (`numeroFK`,`letteraFK`,`scuolaFK`),
  KEY `fk_Iscrizione_Persona1_idx` (`CFFK`),
  KEY `fk_Iscrizione_Anno1` (`inizioFK`,`fineFK`),
  CONSTRAINT `fk_Iscrizione_Anno1` FOREIGN KEY (`inizioFK`, `fineFK`)
REFERENCES `anno` (`inizio`, `fine`),
  CONSTRAINT `fk_Iscrizione_Classe1` FOREIGN KEY (`numeroFK`, `letteraFK`,
`scuolaFK`) REFERENCES `classe` (`numero`, `lettera`, `scuolaFK`),
  CONSTRAINT `fk_Iscrizione_Persona1` FOREIGN KEY (`CFFK`) REFERENCES
`persona` (`CF`);

CREATE TABLE `oresettimanali` (
  `CFFK` char(16) NOT NULL,
  `numeroFK` int NOT NULL,
  `letteraFK` char(1) NOT NULL,
  `scuolaFK` int NOT NULL,
  `inizioFK` int NOT NULL,
  `fineFK` int NOT NULL,
  `oreSettimanali` int unsigned NOT NULL,
  PRIMARY KEY (`CFFK`,`numeroFK`,`letteraFK`,`scuolaFK`,`inizioFK`,`fineFK`),
  KEY `fk_OreSettimanali_Persona1_idx` (`CFFK`),
  KEY `fk_OreSettimanali_Classe1_idx` (`numeroFK`,`letteraFK`,`scuolaFK`),
  KEY `fk_OreSettimanali_Anno1` (`inizioFK`,`fineFK`),
  CONSTRAINT `fk_OreSettimanali_Anno1` FOREIGN KEY (`inizioFK`, `fineFK`)
REFERENCES `anno` (`inizio`, `fine`),
  CONSTRAINT `fk_OreSettimanali_Classe1` FOREIGN KEY (`numeroFK`, `letteraFK`,
`scuolaFK`) REFERENCES `classe` (`numero`, `lettera`, `scuolaFK`),
  CONSTRAINT `fk_OreSettimanali_Persona1` FOREIGN KEY (`CFFK`) REFERENCES
`persona` (`CF`);

CREATE TABLE `parentela` (
  `genitore` char(16) NOT NULL,
  `figlio` char(16) NOT NULL,
  PRIMARY KEY (`genitore`,`figlio`),
  KEY `fk_Persona_has_Persona_Persona1_idx` (`figlio`),
  KEY `fk_Persona_has_Persona_Persona_idx` (`genitore`),
  CONSTRAINT `fk_Persona_has_Persona_Persona` FOREIGN KEY (`genitore`)
REFERENCES `persona` (`CF`),
  CONSTRAINT `fk_Persona_has_Persona_Persona1` FOREIGN KEY (`figlio`)
REFERENCES `persona` (`CF`);
```

```
CREATE TABLE `persona` (
  `CF` char(16) NOT NULL,
  `nome` varchar(50) NOT NULL,
  `cognome` varchar(50) NOT NULL,
  `genere` set('m','f') DEFAULT NULL,
  `telefono` char(10) DEFAULT NULL,
  `dataNascita` date DEFAULT NULL,
  `tipologia` set('alunno','genitore','professore') DEFAULT NULL,
  PRIMARY KEY (`CF`),
  UNIQUE KEY `telefono_UNIQUE` (`telefono`);

CREATE TABLE `plesso` (
  `idPlesso` int NOT NULL AUTO_INCREMENT,
  `nomePlesso` varchar(50) NOT NULL,
  `indirizzo` varchar(100) NOT NULL,
  `città` varchar(50) NOT NULL,
  PRIMARY KEY (`idPlesso`);

CREATE TABLE `scuola` (
  `idScuola` int NOT NULL AUTO_INCREMENT,
  `nomeScuola` varchar(50) NOT NULL,
  `grado` set('asilo','elementare','media') NOT NULL,
  PRIMARY KEY (`idScuola`);

CREATE TABLE `scuolaplesso` (
  `scuolaFK` int NOT NULL,
  `plessoFK` int NOT NULL,
  PRIMARY KEY (`scuolaFK`,`plessoFK`),
  UNIQUE KEY `scuolaFK_UNIQUE` (`scuolaFK`),
  KEY `fk_Scuola_has_Plesso_Plesso1_idx` (`plessoFK`),
  KEY `fk_Scuola_has_Plesso_Scuola1_idx` (`scuolaFK`),
  CONSTRAINT `fk_Scuola_has_Plesso_Plesso1` FOREIGN KEY (`plessoFK`)
REFERENCES `plesso` (`idPlesso`),
  CONSTRAINT `fk_Scuola_has_Plesso_Scuola1` FOREIGN KEY (`scuolaFK`)
REFERENCES `scuola` (`idScuola`);

CREATE TABLE `anno` (
  `inizio` int NOT NULL,
  `fine` int NOT NULL,
  PRIMARY KEY (`inizio`,`fine`);
```

```
CREATE TABLE `aula` (
  `idAula` int NOT NULL AUTO_INCREMENT,
  `mq` int NOT NULL,
  `etàMinima` int NOT NULL,
  `etàMassima` int DEFAULT NULL,
  `capacità` int NOT NULL,
  `plessoFK` int NOT NULL,
  PRIMARY KEY (`idAula`),
  KEY `fk_Aula_Plesso1_idx` (`plessoFK`),
  CONSTRAINT `fk_Aula_Plesso1` FOREIGN KEY (`plessoFK`) REFERENCES `plesso`
(`idPlesso`);

CREATE TABLE `aulaassegnata` (
  `numeroFK` int NOT NULL,
  `letteraFK` char(1) NOT NULL,
  `scuolaFK` int NOT NULL,
  `inizioFK` int NOT NULL,
  `fineFK` int NOT NULL,
  `idAulaFK` int NOT NULL,
  PRIMARY KEY (`numeroFK`,`letteraFK`,`scuolaFK`,`inizioFK`,`fineFK`),
  KEY `fk_AulaAssegnata_Anno1_idx` (`inizioFK`,`fineFK`),
  KEY `fk_AulaAssegnata_Classe1_idx` (`numeroFK`,`letteraFK`,`scuolaFK`),
  KEY `fk_AulaAssegnata_Aula1` (`idAulaFK`),
  CONSTRAINT `fk_AulaAssegnata_Anno1` FOREIGN KEY (`inizioFK`, `fineFK`)
REFERENCES `anno` (`inizio`, `fine`),
  CONSTRAINT `fk_AulaAssegnata_Aula1` FOREIGN KEY (`idAulaFK`) REFERENCES
`aula` (`idAula`),
  CONSTRAINT `fk_AulaAssegnata_Classe1` FOREIGN KEY (`numeroFK`, `letteraFK`,
`scuolaFK`) REFERENCES `classe` (`numero`, `lettera`, `scuolaFK`);

CREATE TABLE `classe` (
  `numero` int NOT NULL,
  `lettera` char(1) NOT NULL,
  `scuolaFK` int NOT NULL,
  PRIMARY KEY (`numero`,`lettera`,`scuolaFK`),
  KEY `fk_Classe_Scuola1_idx` (`scuolaFK`),
  CONSTRAINT `fk_Classe_Scuola1` FOREIGN KEY (`scuolaFK`) REFERENCES
`scuola` (`idScuola`);
```

```
CREATE TABLE `iscrizione` (
  `CFFK` char(16) NOT NULL,
  `numeroFK` int NOT NULL,
  `letteraFK` char(1) NOT NULL,
  `scuolaFK` int NOT NULL,
  `inizioFK` int NOT NULL,
  `fineFK` int NOT NULL,
  PRIMARY KEY (`CFFK`,`numeroFK`,`letteraFK`,`fineFK`,`inizioFK`,`scuolaFK`),
  KEY `fk_Iscrizione_Classe1_idx` (`numeroFK`,`letteraFK`,`scuolaFK`),
  KEY `fk_Iscrizione_Persona1_idx` (`CFFK`),
  KEY `fk_Iscrizione_Anno1` (`inizioFK`,`fineFK`),
  CONSTRAINT `fk_Iscrizione_Anno1` FOREIGN KEY (`inizioFK`, `fineFK`)
REFERENCES `anno` (`inizio`, `fine`),
  CONSTRAINT `fk_Iscrizione_Classe1` FOREIGN KEY (`numeroFK`, `letteraFK`,
`scuolaFK`) REFERENCES `classe` (`numero`, `lettera`, `scuolaFK`),
  CONSTRAINT `fk_Iscrizione_Persona1` FOREIGN KEY (`CFFK`) REFERENCES
`persona` (`CF`);

CREATE TABLE `oresettimanali` (
  `CFFK` char(16) NOT NULL,
  `numeroFK` int NOT NULL,
  `letteraFK` char(1) NOT NULL,
  `scuolaFK` int NOT NULL,
  `inizioFK` int NOT NULL,
  `fineFK` int NOT NULL,
  `oreSettimanali` int unsigned NOT NULL,
  PRIMARY KEY (`CFFK`,`numeroFK`,`letteraFK`,`scuolaFK`,`inizioFK`,`fineFK`),
  KEY `fk_OreSettimanali_Persona1_idx` (`CFFK`),
  KEY `fk_OreSettimanali_Classe1_idx` (`numeroFK`,`letteraFK`,`scuolaFK`),
  KEY `fk_OreSettimanali_Anno1` (`inizioFK`,`fineFK`),
  CONSTRAINT `fk_OreSettimanali_Anno1` FOREIGN KEY (`inizioFK`, `fineFK`)
REFERENCES `anno` (`inizio`, `fine`),
  CONSTRAINT `fk_OreSettimanali_Classe1` FOREIGN KEY (`numeroFK`, `letteraFK`,
`scuolaFK`) REFERENCES `classe` (`numero`, `lettera`, `scuolaFK`),
  CONSTRAINT `fk_OreSettimanali_Persona1` FOREIGN KEY (`CFFK`) REFERENCES
`persona` (`CF`);

CREATE TABLE `parentela` (
  `genitore` char(16) NOT NULL,
  `figlio` char(16) NOT NULL,
  PRIMARY KEY (`genitore`,`figlio`),
  KEY `fk_Persona_has_Persona_Persona1_idx` (`figlio`),
  KEY `fk_Persona_has_Persona_Persona_idx` (`genitore`),
  CONSTRAINT `fk_Persona_has_Persona_Persona` FOREIGN KEY (`genitore`)
REFERENCES `persona` (`CF`),
  CONSTRAINT `fk_Persona_has_Persona_Persona1` FOREIGN KEY (`figlio`)
REFERENCES `persona` (`CF`);
```

```
CREATE TABLE `persona` (
  `CF` char(16) NOT NULL,
  `nome` varchar(50) NOT NULL,
  `cognome` varchar(50) NOT NULL,
  `genere` set('m','f') DEFAULT NULL,
  `telefono` char(10) DEFAULT NULL,
  `dataNascita` date DEFAULT NULL,
  `tipologia` set('alunno','genitore','professore') DEFAULT NULL,
  PRIMARY KEY (`CF`),
  UNIQUE KEY `telefono_UNIQUE` (`telefono`);

CREATE TABLE `plesso` (
  `idPlesso` int NOT NULL AUTO_INCREMENT,
  `nomePlesso` varchar(50) NOT NULL,
  `indirizzo` varchar(100) NOT NULL,
  `città` varchar(50) NOT NULL,
  PRIMARY KEY (`idPlesso`);

CREATE TABLE `scuola` (
  `idScuola` int NOT NULL AUTO_INCREMENT,
  `nomeScuola` varchar(50) NOT NULL,
  `grado` set('asilo','elementare','media') NOT NULL,
  PRIMARY KEY (`idScuola`);

CREATE TABLE `scuolaplesso` (
  `scuolaFK` int NOT NULL,
  `plessoFK` int NOT NULL,
  PRIMARY KEY (`scuolaFK`,`plessoFK`),
  UNIQUE KEY `scuolaFK_UNIQUE` (`scuolaFK`),
  KEY `fk_Scuola_has_Plesso_Plesso1_idx` (`plessoFK`),
  KEY `fk_Scuola_has_Plesso_Scuola1_idx` (`scuolaFK`),
  CONSTRAINT `fk_Scuola_has_Plesso_Plesso1` FOREIGN KEY (`plessoFK`)
REFERENCES `plesso` (`idPlesso`),
  CONSTRAINT `fk_Scuola_has_Plesso_Scuola1` FOREIGN KEY (`scuolaFK`)
REFERENCES `scuola` (`idScuola`);
```

5.2. Definizione delle interrogazioni per la visualizzazione dei dati

QUERY 1 *Inserire una nuova iscrizione.*
INSERT INTO iscrizione VALUES ('CFFK', 'NUMERO AULA', LETTERA AULA', 'NUMERO SCUOLA', 'ANNO INIZIO', 'ANNO FINE')

Esempio: INSERT INTO iscrizione VALUES (VLADNL99R14D086C, '1, A, '1, '2020', '2021')

QUERY 2 *Trovare gli studenti iscritti alla scuola secondaria inferiore nell'anno 2020.*

```
SELECT  persona.*
FROM    persona, iscrizione, scuola
WHERE   persona.CF = iscrizione.CFFK AND
        iscrizione.scuolaFK = scuola.idScuola AND
        scuola.grado = 'media' AND
        iscrizione.inizioFK = '2020';
```

QUERY 3 *Trovare gli studenti che hanno frequentato almeno una classe per ogni grado dell'istituto.*

```
SELECT  CF
FROM    persona
WHERE   CF IN(SELECT CF FROM iscrizioniasilo) AND
        CF IN(SELECT CF FROM iscrizionielementari) AND
        CF IN(SELECT CF FROM iscrizionimedia);
```

QUERY 4 *Trovare i plessi con il numero di studenti ospitabili più alto.*

```
SELECT  studentiospitabili.idPlesso, studentiospitabili.capacita
FROM    studentiospitabili
WHERE   studentiospitabili.capacita  >=
        ALL(SELECT studentiospitabili.capacita FROM studentiospitabili);
```

QUERY 5 *Trovare, per ogni anno, i gradi con il maggior numero di iscritti.*

```
SELECT  anno, scuola
FROM    infoscuola as f
WHERE   f.iscrittiTOT >=
        ALL(SELECT x.iscrittiTOT FROM infoscuola as x WHERE x.anno = f.anno);
```

QUERY 6 *Controllare che ad un'aula non vengano assegnati più iscritti di quanti ne possa ospitare.*
Vedi Trigger "Verifica correttezza iscrizione"

QUERY 7 *Alla scadenza delle iscrizioni annuali (1 febbraio) si memorizzi il numero di nuovi iscritti per grado e classe (1-4 infanzia, 1-5 primaria e 1-3 medie).*
Vedi Evento "Iscritti_totali"

5.3. Trigger e Procedure schedulate

Si assicura che la nuova iscrizione rispetti i vincoli. Una iscrizione è corretta quando:

1. *L'età dello studente è coerente con l'età minima della classe al quale viene iscritto*
2. *Lo studente deve essere memorizzato come alunno*
3. *Le iscrizioni avvengono solo nel mese di gennaio*
4. *Lo studente può iscriversi solo all'anno successivo a quello corrente*
5. *L'aula associata alla classe non deve essere piena*

```
CREATE DEFINER=`root`@`localhost` TRIGGER `Verifica correttezza iscrizione` BEFORE INSERT ON
`iscrizione` FOR EACH ROW BEGIN
	declare etaStudente int;
	declare minimoEta int;
	declare max int;
	declare size int;

    set etaStudente = year(current_date()) - (SELECT year(dataNascita) FROM persona WHERE
persona.CF = new.CFFK);

	set minimoEta = (SELECT etàMinima FROM aula, aulaassegnata as X WHERE idAula =
X.idAulaFK AND X.numeroFK = new.numeroFK AND X.letteraFK = new.letterafk AND X.scuolaFK =
new.scuolaFK AND X.inizioFK = new.inizioFK AND X.fineFK = new.fineFK);

	if( etaStudente < minimoEta ) then
	signal sqlstate '45000' set message_text = 'Alunno troppo piccolo per essere iscritto a questa
classe';
	end if;
	if ( (SELECT tipologia FROM persona WHERE CF = new.CFFK) != 'alunno' ) then
	signal sqlstate '45000' set message_text = 'Si possono iscrivere solo alunni';
	end if;
	if ( month(current_date()) != 01 ) then
	signal sqlstate '45000' set message_text = 'Iscrizioni non aperte';
	end if;
	 if( year(current_date()) != new.inizioFK ) then
	signal sqlstate '45000' set message_text = 'Impossibile iscrivere a questo anno scolastico';
	end if;

	set size = (SELECT count(*) FROM iscrizione WHERE inizioFK = new.inizioFK AND numeroFK =
new.numeroFK AND letteraFK = new.letteraFK AND scuolaFK = new.scuolaFK);

	 set max = (SELECT capacità FROM aula, aulaassegnata WHERE idAula = idAulaFK AND
scuolaFK = new.scuolaFK AND letteraFK = new.letteraFK AND numeroFK = new.numeroFK);

	if( size+1 > max) then
	signal sqlstate '45000' set message_text = 'Impossibile iscrivere, capienza massima aula
raggiunta';
	end if;
END
```

Si assicura che la classe venga numerata rispettando i vincoli.

```
CREATE DEFINER=`root`@`localhost` TRIGGER `Numerazione classi corretta` BEFORE INSERT ON `classe`
FOR EACH ROW BEGIN
	if ((SELECT grado FROM scuola WHERE new.scuolaFK = idScuola)='asilo' AND new.numero>4) then
	signal sqlstate '45000' set message_text = 'Numero max classi asilo = 4';
	end if;

if ((SELECT grado FROM scuola WHERE new.scuolaFK=idScuola)='elementare' AND new.numero>5)then
	signal sqlstate '45000' set message_text = 'Numero max classi elementare = 5';
	end if;

	 if ((SELECT grado FROM scuola WHERE new.scuolaFK=idScuola)='media' and new.numero>3) then
	signal sqlstate '45000' set message_text = 'Numero max classi media = 3';
	end if;
END
```

Si assicura che una persona non sia parente di sè stesso.

```
CREATE DEFINER=`root`@`localhost` TRIGGER `Persone diverse` BEFORE INSERT ON `parentela` FOR
EACH ROW BEGIN
   if (new.genitore = new.figlio) then
   signal sqlstate '45000' set message_text = 'Non si può inserire la stessa persona';
   end if;
END
```

Il nuovo anno scolastico viene inserito automaticamente.

```
delimiter //
CREATE EVENT Anno_Scolastico ON SCHEDULE EVERY 1 year STARTS '2016-01-01 00:00:00' DO
BEGIN
	INSERT INTO anno VALUES (year(current_timestamp()), year(current_timestamp())+1);
END//
delimiter ;
```

Crea un report annuale contenente le nuove iscrizioni

```
delimiter //
CREATE EVENT Iscritti_totali ON SCHEDULE EVERY 1 year starts '2017-02-01 00:00:00' DO
BEGIN
	CREATE OR REPLACE VIEW  iscritti_Anno (Numero, Lettera, grado, num_Alunni) AS
	SELECT		numeroFK, letteraFK, grado, count(*)
	FROM		iscrizione, scuola
	WHERE		scuolaFK = idScuola AND inizioFK = year(now())
	GROUP BY	numeroFK, letteraFK, grado
	ORDER BY	grado;
END//
delimiter ;
```

5.4. Script SQL-DML : analisi puntuali

```
CREATE OR REPLACE VIEW studentiOspitabili ( idPlesso, capacita ) AS
SELECT      idPlesso, SUM(capacità)
FROM        aula, plesso
WHERE       aula.plessoFK = plesso.idPlesso
GROUP BY    idPlesso;

CREATE OR REPLACE VIEW infoScuola(scuola, anno, iscrittiTOT) AS
SELECT      scuolaFK, inizioFK, count(*)
FROM        iscrizione
GROUP BY    scuolaFK, inizioFK;

CREATE OR REPLACE VIEW iscritti_Anno (Numero, Lettera, grado, num_Alunni) AS
SELECT      numeroFK, letteraFK, grado, count(*)
FROM        iscrizione, scuola
WHERE       scuolaFK = idScuola AND
            inizioFK = year(now())
GROUP BY numeroFK, letteraFK, grado
ORDER BY grado;

CREATE OR REPLACE VIEW iscrizioniasilo (CF) AS
SELECT      CFFK
FROM        iscrizioni, scuola
WHERE       scuolaFK = idScuola AND
            grado = 'asilo';

CREATE OR REPLACE VIEW iscrizionielementari (CF) AS
SELECT      CFFK
FROM        iscrizioni, scuola
WHERE       scuolaFK = idScuola AND
            grado = 'elementari';

CREATE OR REPLACE VIEW iscrizionimedia (CF) AS
SELECT      CFFK
FROM        iscrizioni, scuola
WHERE       scuolaFK = idScuola AND
            grado = 'media';
```

5.5. Script SQL-DML : analisi aggregate

```
INSERT INTO iscrizione VALUES ('CFFK', 'NUMERO AULA', LETTERA AULA', 'NUMERO SCUOLA', 'ANNO INIZIO', 'ANNO FINE')
```