Progettazione di una base di dati per la gestione condominiale

Raffaele Sommese, Antonio Pizzata

20 gennaio 2015

A.1 INTRODUZIONE

Questo documento presenta un esempio di progettazione di basi di dati riguardante la gestione di un condominio.
L'obiettivo della base di dati è quello di realizzare un sistema informatico che faciliti la gestione delle informazioni e delle attività legate alla memorizzazione delle spese, dei pagamenti effettuati, ricevuti e ancora da ricevere.

A.2 SPECIFICHE SUI DATI

Ci si propone di realizzare una base di dati per la gestione di un condominio. Essa conterrà le informazioni relative ai condomini, alle spese ed ai pagamenti.
Per ogni condomino si vuole conoscere l'interno (unico), il nome, il cognome, il telefono e la scala a cui appartiene. Ad ogni condomino sono associate, poi, più tabelle millesimali e al massimo due quote mensili da pagare. Delle tabelle si vuole conservare il nome ed i millesimi corrispondenti. Delle quote, invece, l'importo, la data, la modalità di pagamento, l'anno ed il mese a cui si riferiscono.
La base dovrà inoltre contenere le informazioni relative alle spese: l'anno, l'importo e la causale. Esse sono anche divise tra effettive e preventivate; le prime, sono caratterizzate da una modalità di pagamento, da una data di evasione e dall'essere straordinarie oppure ordinarie.
Per ogni spesa preventivata, inserita all'inizio dell'anno, il sistema genererà le singole quote mensili per ogni condomino. Nel caso di spese straordinarie, in seguito al loro inserimento nella base di dati, verrano suddivise tra i vari condomini secondo le relative tabelle millesimali. Ogni volta che viene ricevuto un pagamento, verrà poi memorizzata la data di evasione del medesimo.

A.2.1 SPECIFICHE SULLE OPERAZIONI

Riportiamo ora le operazioni che possono essere effettuate sulla base di dati.

• Aggiornamento quote: suddivide le spese effettive/straordinarie tra i vari condomini (ogni volta che viene inserita una spesa).

• Inserimento/cancellazione condomini: inserisce/cancella un condomino dalla base di dati.

• Inserimento/cancellazione spese: inserisce/cancella una spesa dalla base di dati.

• Generamento delle quote: dalle spese preventivate si ottengono le quote che il singolo condomino dovrà pagare (una volta l'anno).

• Stampa delle quote.

• Inserimento dei millesimi: inserisce i millesimi e la relativa tabella millesimale in corrispondenza di un condomino (ogni qualvolta viene inserito un condomino).

• Controllo di consistenza dei millesimi: operazione che somma tutti i millesimi dei condomini e controlla se il risultato è 1000.

• Gestione degli anni: incrementa di 1 l'anno corrente.

A.2.2 ANALISI DELLE SPECIFICHE E RISTRUTTURAZIONE DEI REQUISITI

Informazioni generali

Si vuole progettare una base di dati per la gestione di un condominio che contenga informazioni relative ai condomini, ai loro millesimi e alle spese.

Informazioni sui condomini

Dei condomini si vogliono conservare nome, cognome, telefono, interno (unico per ognuno di essi) e scala.

Informazioni sulle tabelle millesimali

Delle tabelle millesimali si vuole conoscere la tabella ed il relativo millesimo.

Informazioni sulle spese

La base di dati deve tener traccia di tutte le spese, preventivate ed effettive, a cui va incontro il condominio. Ogni spesa è caratterizzata da una causale e da un importo. Delle spese effettive si vuole conservare la data e l'essere straordinaria o ordinaria.

Informazioni sulle quote

Per ciò che interessa le quote appartenenti al singolo condomino, la base di dati conterrà informazioni riguardo l'anno, il mese, l'importo, l'essere pagate o non pagate e delle note in cui si specifica il motivo di tale spesa.

Informazioni sugli anni e sui mesi
La base di dati deve conservare le varie annualità e i mesi. Per i primi si vuole sapere l'anno e se esso si riferisce alla gestione corrente o meno. Per i secondi si vuole sapere il mese ed il suo codice univoco.
Informazioni sui metodi di pagamento
Riguardo i metodi di pagamento si vogliono salvare nella base di dati un ID del pagamento, unico, e la relativa modalità di pagamento.

A.2.3. PROGETTAZIONE CONCETTUALE
Schema ER portante
Lo schema ER portante è il seguente:

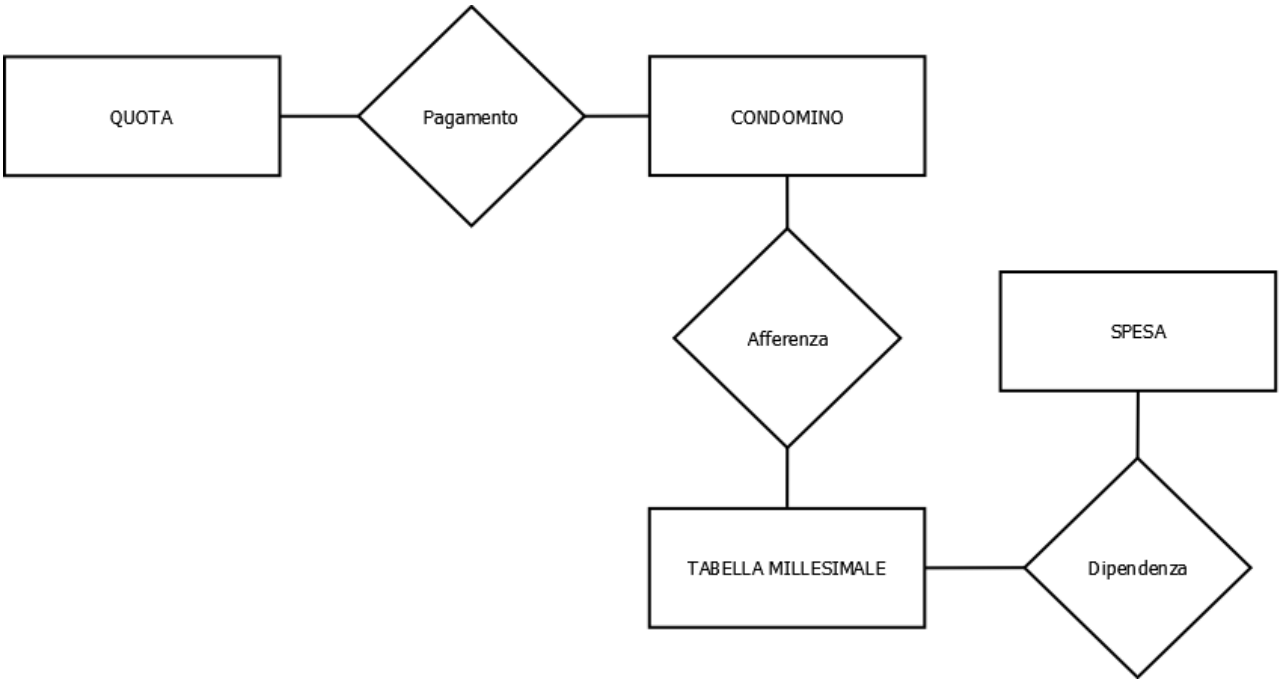

Figura 1: ER Portante

Come si può ben vedere, si evidenziano quattro entità fondamentali: *Quota, Condomino, Tabella millesimale* e *Spesa*.
Queste rappresentano rispettivamente: la spesa a cui va incontro un condomino, il proprietario di un immobile, la tabella millesimale legata ad un condomino e la spesa necessaria per il mantenimento dello stabile.

Le entità *Quota* e *Condomino* sono legate dalla relazione *Pagamento*; le entità *Condomino* e *Tabella millesimale* sono connesse tra loro grazie alla relazione *Afferenza*; le entità *Tabella millesimale* e *Spesa* sono legate dalla relazione *Dipendenza*.

Termine	**Descrizione**	**Termini Collegati**
Condomino	Proprietario di un appartamento o di un locale in un condominio	Quota, Tabella millesimale, Pagamento
Tabella millesimale	Rappresenta la quota di proprietà nel condominio, espressa come rapporto fra il valore di ciascuna unità e il valore dell'intero edificio, fatto uguale a 1.000	Condomino, Spesa
Quota	Somma che ciascun condomino deve pagare	Condomino, Spesa
Spesa preventivata	Spesa prevista con relativo importo, il cui totale corrisponderà alla somma complessiva che sarà riscossa durante l'anno cui si riferisce	Tabella millesimale, Condomino, Anno
Spesa effettiva	Prospetto nel quale vengono ripartite le spese a tutte le unità immobiliari secondo le tabelle millesimali	Tabella millesimale, Condomino

Tabella 1: **Tabella A.2.3** Glossario dei termini

Poichè ad ogni *Condomino* corrispondono più *Tabelle Millesimali*, esso partecipa alla relazione *Afferenza* con cardinalità (1,N). L'entità *Tabella Millesimale* di tale associazione vi partecipa con cardinalità (1,1). Le stesse cardinalità le ritroviamo tra *Condomino* e *Quota* nella relazione *Pagamento* e tra *Tabella millesimale* e *Spesa* nella relazione *Dipendenza*.

Schema ER finale

Analizzando meglio le specifiche ci si rende conto che vi sono altre entità da aggiungere. Ad esempio: poichè una data spesa, o una data quota, sono localizzate temporalmente è utile aggiungere le entità ***Anno*** e ***Mese***. La

prima ha come attributi Anno (univoco) e InCorso, che indica se l'anno preso in considerazione è corrente. La seconda possiede, invece, gli attributi DescrizioneMese, che indica di che mese si tratta, e CodMese, codice univoco associato ad ogni mese.
Altra entità necessaria per il corretto funzionamento della base di dati è ***ModPagamento***, la quale è caratterizzata da due attributi: Pagamento, che indica il tipo di pagamento, e IdPagamento, univoco, che è un codice associato alla modalità di pagamento. Come si può ben immaginare, a tali entità vengono associate altre relazioni con le relative cardinalità:
•Un *Anno* può essere collegato a più *Quote* o *Spese* oppure a nessuna di esse tramite le relazioni *Gestione1* e *Gestione2* (cardinalità: (0,N)). Una spesa o una quota si riferiscono, invece, ad un unico *Anno* (cardinalità: (1,1)).
•L'entità *Mese* è collegata all'entità *Quota* tramite la relazione *Individuazione*. Quest'ultima indica che un *Mese* può avere da 0 a N *Quote* (cardinalità: (0,N)) e che ad ogni *Quota* corrisponde uno ed un solo *Mese* (cardinalità: (1,1)).
•L'entità *ModPagamento* è collegata all'entità *Quota* e all'entità *Spesa Effettiva* tramite le relazioni *Appartenenza1* e *Appartenenza2*, rispettivamente. Queste relazioni mostrano che una *Spesa* ed una *Quota* possono avere un'unica modalità di pagamento (carinalità :(1,1)), mentre ogni modalità di pagamento può avere da 0 a N *Spese* o *Quote* a cui essere associata (cardinalità: (0,N)).
•L'entità *Spesa* è caratterizzata dagli attributi Causale e Importo. Essa è, poi, padre di altre due entità: *Spesa Preventivata* e *Spesa Effettiva*. La prima eredita tutti gli attributi del padre, mentre la seconda si specializza aggiungendo l'attributo Data, unico, e Ord/Straord, che indica se la spesa è ordinaria oppure straordinaria.
La Figura 2 mostra il modello ER concettuale complessivo.

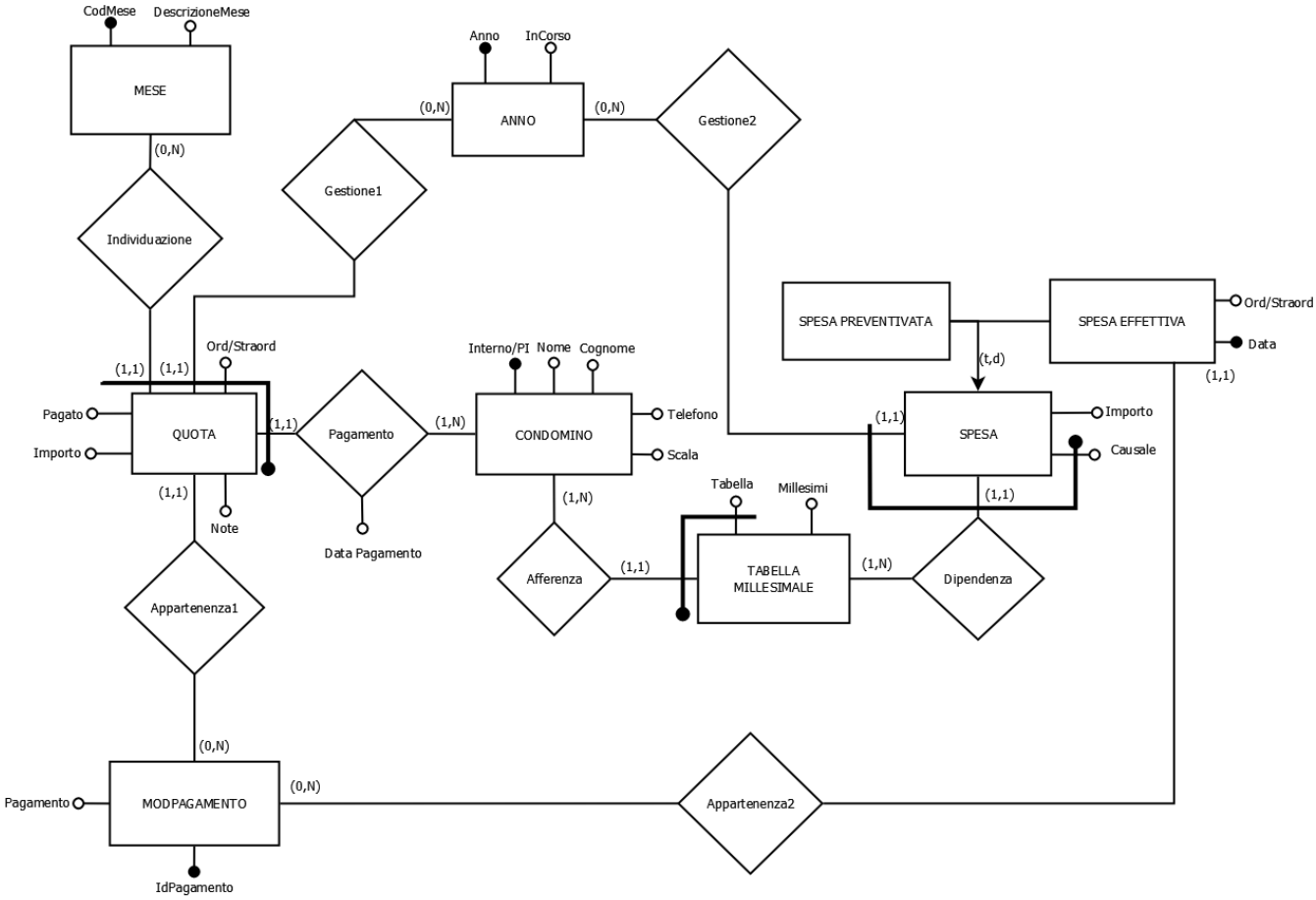

Figura 2: ER concettuale complessivo

A.2.4 PROGETTAZIONE LOGICA

Fase 1: Trasformazione

Con riferimento alla Figura 2, la trasformazione dello schema concettuale consisterà nell'eliminazione della generalizzazione *Spesa*. Per tale scopo uccidiamo il padre, aggiungendo così le due entità *Spesa Preventivata* e *Spesa Effettiva*, alle quali vengono asseganti gli attributi del padre e le relative associazioni (*Dipendenza1* e *Dipendenza2*).
I motivi di tale scelta sono vari:

• *Il tipo di generalizzazione*: *totale* e *disgiunta*.
• *La tipologia delle operazioni sulle entità figlie*: le operazioni sulla base di dati accedono ad occorrenze del padre e di almeno una figlia, essendo poi le operazioni diverse tra di loro a seconda dell'entità a cui si riferiscono, si preferisce scegliere questa soluzione.

Fase 2: Traduzione

• Ogni entità si trasforma in una relazione avente come attributi quelli dell'entità e come chiavi primarie i suoi identificatori.
• Essendo tutte le relazioni *uno a molti* esse scompaiono e, contemporaneamente, gli identificatori dal lato *molti* passano all'entità lato *uno*. Per esempio, le entità *Quota* e *Condomino* sono collegate tra di loro dalla relazione *Pagamento*.

Di seguito è riportato il modello ER trasformato della base di dati in esame.

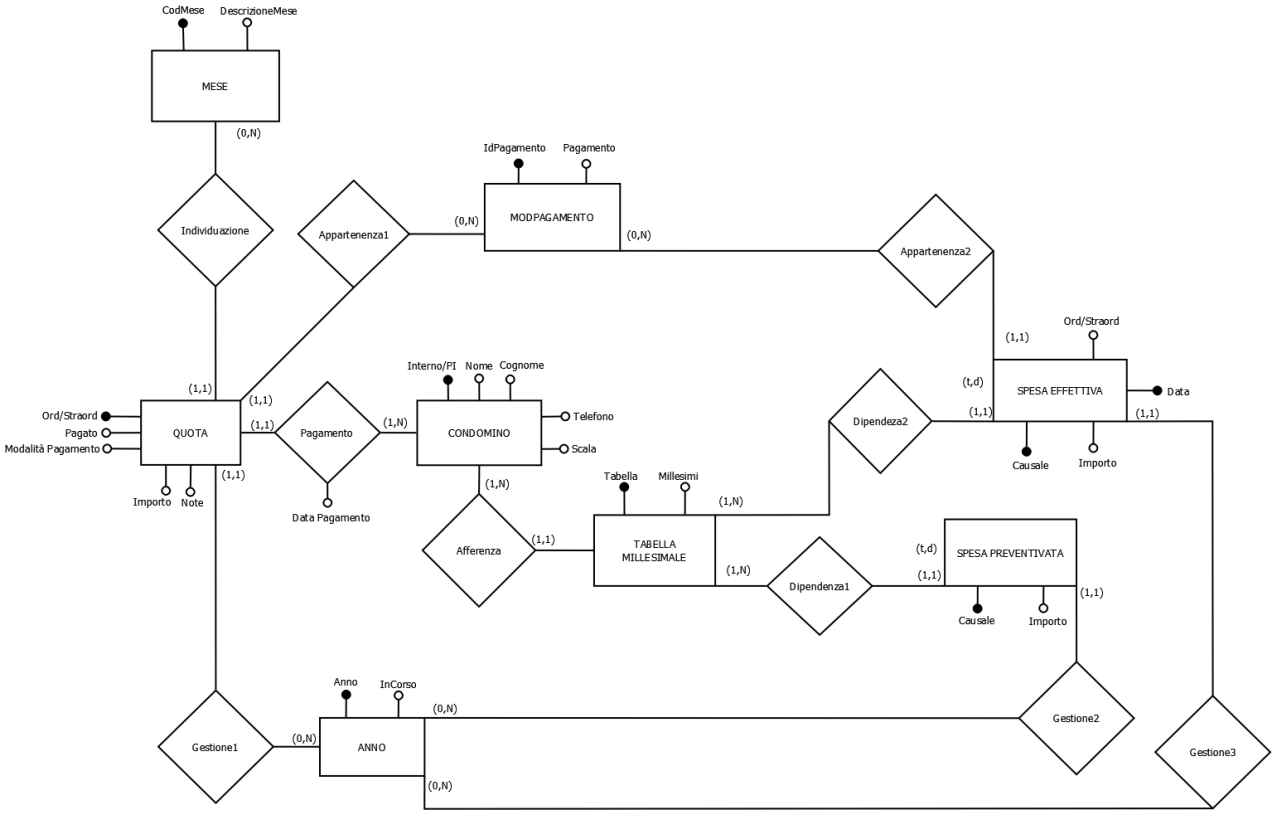

Figura 3: ER Trasformato

Di seguito è riportato lo schema relazionale completo della base di dati in esame.
CONDOMINI (Interno/PI, Nome, Cognome, Telefono, Scala)
QUOTE (Ord/Straod, Interno/PI:CONDOMINI, Anno:ANNI, Mese:MESI, Modalità Pagamento:MODPAGAMENTO, Pagato)
TABELLE MILLESIMALI (Tabella, Interno/PI:CONDOMINI, millesimi)
SPESE PREVENTIVATE (Causale, Anno:ANNI, Tabella:TABELLE MILLESIMALI, ModalitàPagamento:MODPAGAMENTO, Importo)
SPESE EFFETTIVE (Causale, Data, Anno:ANNI, Tabella:TABELLE MILLESIMALI, ModalitàPagamento:MODPAGAMENTO, Importo, OrdStraord)
ANNI (Anno, InCorso)
MESI (CodMese, DescrizioneMese)
MODPAGAMENTO (IdPagamento, Pagamento)

A.2.5 PROGETTAZIONE FISICA

Condizioni operative:
• Utilizzo del database MySql.
• Server con cpu Intel® Xeon® Processor E7-4890 v2 (37.5M Cache, 2.80 GHz), 32 GB di RAM, 4 HD 146 GB SAS (in RAID 5 con 438 GB di memoria disponibile).
• Sistema operativo del Server: Debian 8 Jessie.

Dimensionamento fisico della base di dati
Prima dello studio del dimensionamento fisico della base vogliamo ricordare l'occupazione in termini di byte del DBMS MySql dei tipi utilizzati:

•Tinyint: 1 byte.
•Int,Integer: 4 bytes.
•Decimal: sono rappresentati utilizzando un formato binario che racchiude nove cifre decimali (base 10) in quattro byte. Le parti intere e quelle decimali di ogni valore sono determinate separatamente. Ogni multiplo delle nove cifre richiede quattro byte, e le cifre rimanenti richiedono una frazione dei quattro byte. La memoria richiesta per le cifre in eccesso è data dalla seguente tabella:

Cifre rimanenti	Numero di Bytes
0	0
1	1
2	1
3	2
4	2
5	3
6	3
7	4
8	4

•Varchar(m): da 0 a m + 1 bytes se la stringa necessita da 0 a 255 bytes, altrimenti da 0 a m + 2 bytes per stringhe da più di 255 bytes.
•Text: L + 2 bytes, dove L $< 2^{16}$.
•Date: 3 bytes. Riportiamo, di seguito,una stima dell'occupazione di memoria a regime, all'anno della messa in esercizio (initial), e quella prevista nel successivo biennio di funzionamento (next).

Attributo	**Tipo**	**Byte**	**Initial**	**Next**
			50 occ.	100 occ.
Interno/PI	VARCHAR(5)	6	300 b	600 b
Nome	VARCHAR(45)	46	2,3 K	4,6 K
Cognome	VARCHAR(45)	46	2,3 K	4,6 K
Telefono	INT(11)	4	200 b	400 b
Scala	VARCHAR(1)	2	100 b	200 b
Totale (dati)			5,2 K	10,4 K

Tabella 1: **Dimensionamento Tabella *CONDOMINI***

Attributo	**Tipo**	**Byte**	**Initial**	**Next**
			800 occ.	2000 occ.
Mese	INT(2)	4	3,2 K	8 K
Anno	INT(4)	4	3,2 K	8 K
Pagato	TINYINT(1)	1	800 b	2 K
Importo	DECIMAL(10,2)	6	4,8 K	12 K
Note	TEXT	1026	820 K	2,052 M
Interno/PI	VARCHAR(5)	6	4,8 K	12 K
DataPagamento	DATE	3	2,4 K	6 K
ModoPagamento	VARCHAR(1)	2	1,6 K	4 K
OrdStraord	TINYINT(1)	1	800 b	2 K
Totale (dati)			841,6 K	2,106 M

Tabella 2: **Dimensionamento Tabella *QUOTE***

Attributo	**Tipo**	**Byte**	**Initial**	**Next**
			200 occ.	400 occ.
Anno	INT(4)	4	800 b	1,6 k
Causale	VARCHAR(255)	256	51,2 k	102,4 K
Data	DATE	3	600 b	1,2 K
Importo	DECIMAL(10,2)	6	1,2 K	2,4 K
ModoPagamento	VARCHAR(1)	2	400 b	800 b
Tabella	VARCHAR(1)	2	400 b	800 b
OrdStraord	TINYINT(1)	1	200 b	400 b
Totale (dati)			54,8 K	109,6 K

Tabella 3: **Dimensionamento Tabella *SPESE EFFETTIVE***

Attributo	**Tipo**	**Byte**	**Initial**	**Next**
			200 occ.	400 occ.
Anno	INT(4)	4	800 b	1,6 k
Causale	VARCHAR(255)	256	51,2 k	102,4 K
Importo	DECIMAL(10,2)	6	1,2 K	2,4 K
Tabella	VARCHAR(1)	2	400 b	800 b
Totale (dati)			53,6 K	104,2 K

Tabella 4: **Dimensionamento Tabella *SPESE PREVENTIVATE***

Attributo	**Tipo**	**Byte**	**Initial**	**Next**
			250 occ.	500 occ.
Condomino	VARCHAR(5)	6	1,5 K	3 K
Tabella	VARCHAR(1)	2	500 B	1 K
Millesimi	DECIMAL (6,2)	4	1 K	2 K
Totale (dati)			3 K	6 K

Tabella 5: **Dimensionamento Tabella *TABELLE MILLESIMALI***

Attributo	**Tipo**	**Byte**	**Initial**	**Next**
			12 occ.	1 occ.
CodMese	INT(2)	4	48 b	4 b
DescrizioneMese	VARCHAR(11)	12	144 b	12 b
Totale (dati)			196 b	16 b

Tabella 6: **Dimensionamento Tabella *MESI***

Attributo	**Tipo**	**Byte**	**Initial**	**Next**
			4 occ.	2 occ.
Anno	INT(4)	4	16 b	8 b
InCorso	TINYINT(1)	1	4 b	2 b
Totale (dati)			20 b	10 b

Tabella 7: **Dimensionamento Tabella *ANNI***

Attributo	**Tipo**	**Byte**	**Initial**	**Next**
			6 occ.	2 occ.
CodPagamento	VARCHAR(1)	2	12 b	4
DescrPagamento	VARCHAR(15)	16	96 b	36
Totale (dati)			108 b	40 b

Tabella 8: **Dimensionamento Tabella *MODPAGAMENTO***

Si è scelto poi di utilizzare vari indici per rendere più veloce l'accesso alle varie tabelle:
•Tabella *Anni*: INCORSO.IDX
•Tabella *Condomini*: NomeCogn.IDX
•Tabella *Quote*: fkAnnoidx.IDX, fkQuoteCondominiidx.IDX, fkQuoteModopagidx.IDX, Gestione.IDX
•Tabella *Spese Effettive*: anno.IDX, fkSpeseEffettive.IDX, fkSpeseModpaga.IDX, Gestione.IDX
•Tabella *Spese Preventivate*: anno.IDX, fkSpesePreventivateTabellaMillesimale1.IDX
•*Tabelle Millesimale*: fkTabellaMillesimaleCondomini1.IDX, Tabelle.IDX
TUTTE le tabelle del database sono poi dotate di un indice *PRIMARY.IDX*.

La creazione del database
Per la creazione della base di dati si è scelto di usare il DBMS *MySql*, mentre come gestore della memoria si è optato l'utilizzo di *Innodb*. Quest'ultimo permette l'allocazione dinamica di tutte le risorse, in modo da evitare l'utilizzo di tablespaces.

Creazione degli utenti e delle politiche di sicurezza
Riguardo la creazione degli utenti, si è deciso di crearne tre:
• Amministratore: può selezionare e manipolare le tabelle a suo piacimento.
• Condomino: può selezionare solo la propria vista/tabella.
• Dba: ha tutti i privilegi. Di seguito riportiamo gli script:

Utente generico
CREATE USER 'utente'@'%'
IDENTIFIED BY 'some_pass';
Amministratore
GRANT USAGE ON *.* TO 'Amministratore'@'%' IDENTIFIED BY PASSWORD 'some_pass';
GRANT SELECT, INSERT, UPDATE, DELETE, REFERENCES, INDEX, CREATE TEMPORARY TABLES, LOCK TABLES, EXECUTE, SHOW VIEW, EVENT, TRIGGER ON `Condominio_test`.* TO 'Amministratore'@'%';

Condomino
GRANT USAGE ON *.* TO 'h'@'%' IDENTIFIED BY PASSWORD 'some_pass' WITH MAX_QUERIES_PER_HOUR 100 MAX_CONNECTIONS_PER_HOUR 60 MAX_USER_CONNECTIONS 10;
GRANT SELECT ON `Condominio_test`.`QuoteCondomini` TO 'h'@'%';

Dba
GRANT ALL PRIVILEGES ON *.* TO 'dba'@'%' IDENTIFIED BY PASSWORD 'some_pass' WITH GRANT OPTION;

Creazione degli oggetti della base di dati
Tabella **Anni**
DROP TABLE IF EXISTS `Anni`;
CREATE TABLE IF NOT EXISTS `Anni` (
`Anno` INT(4) NOT NULL,
`InCorso` tinyint(1) NOT NULL DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE `Anni`
ADD PRIMARY KEY (`Anno`), ADD KEY `INCORSO` (`InCorso`) USING BTREE;

Tabella **Condomini**
DROP TABLE IF EXISTS `Condomini`;
CREATE TABLE IF NOT EXISTS `Condomini` (
`Interno_PI` VARCHAR(5) NOT NULL,
`Nome` VARCHAR(45) NOT NULL,

‘Cognome‘ VARCHAR(45) NOT NULL,
‘Telefono‘ INT(11) DEFAULT NULL,
‘Scala‘ VARCHAR(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE ‘Condomini‘
ADD PRIMARY KEY (‘Interno_PI‘), ADD KEY ‘NomeCogn‘ (‘Nome‘,‘Cognome‘)
USING BTREE;

Tabella **Mesi**
DROP TABLE IF EXISTS ‘Mesi‘;
CREATE TABLE IF NOT EXISTS ‘Mesi‘ (
‘CodMese‘ INT(2) NOT NULL,
‘DescrizioneMese‘ VARCHAR(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE ‘Mesi‘
ADD PRIMARY KEY (‘CodMese‘);

Tabella **ModoPagamento**
DROP TABLE IF EXISTS ‘ModoPagamento‘;
CREATE TABLE IF NOT EXISTS ‘ModoPagamento‘ (
‘CodPagamento‘ VARCHAR(1) NOT NULL,
‘DescrPagamento‘ VARCHAR(15) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE ‘ModoPagamento‘
ADD PRIMARY KEY (‘CodPagamento‘);

Tabella **Quote**
DROP TABLE IF EXISTS ‘Quote‘;
CREATE TABLE IF NOT EXISTS ‘Quote‘ (
‘Mese‘ INT(2) NOT NULL,
‘Anno‘ INT(4) NOT NULL,
‘Pagato‘ tinyint(1) NOT NULL DEFAULT ’0’,
‘Importo‘ DECIMAL(10,2) NOT NULL,
‘Note‘ text,
‘Interno_PI‘ VARCHAR(5) NOT NULL,

‘Data_Pagamento‘ DATE DEFAULT NULL,
‘Modo_Pagamento‘ VARCHAR(1) DEFAULT NULL,
‘OrdStraord‘ tinyint(1) NOT NULL DEFAULT ’1’
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE ‘Quote‘
ADD PRIMARY KEY (‘Mese‘,‘Anno‘,‘Interno_PI‘,‘OrdStraord‘), ADD KEY
‘fk_Quote_Condomini_idx‘ (‘Interno_PI‘), ADD KEY ‘fk_Anno_idx‘ (‘Anno‘), ADD KEY ‘fk_Quote_modpag_idx‘ (‘Modo_Pagamento‘), ADD KEY
‘Gestione‘ (‘OrdStraord‘) USING BTREE;

Tabella **Spese_Effettive**

DROP TABLE IF EXISTS ‘Spese_Effettive‘;
CREATE TABLE IF NOT EXISTS ‘Spese_Effettive‘ (
‘Anno‘ INT(11) NOT NULL,
‘Causale‘ VARCHAR(255) NOT NULL,
‘Data‘ DATE NOT NULL,
‘Importo‘ DECIMAL(10,2) NOT NULL,
‘Modo_Pagamento‘ VARCHAR(1) DEFAULT NULL,
‘TabellaFK‘ VARCHAR(1) NOT NULL,
‘Ord_Straord‘ tinyint(1) NOT NULL DEFAULT ’0’
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE ‘Spese_Effettive‘
ADD PRIMARY KEY (‘Anno‘,‘Causale‘,‘Data‘), ADD KEY ‘fk_Spese_Effettive_Tabella_Millesi
(‘TabellaFK‘), ADD KEY ‘fk_spese_modpaga_idx‘ (‘Modo_Pagamento‘),
ADD KEY ‘Gestione‘ (‘Ord_Straord‘) USING BTREE, ADD KEY ‘Anno‘
(‘Anno‘) USING BTREE;

Tabella **Spese_Preventivate**

DROP TABLE IF EXISTS ‘Spese_Preventivate‘;
CREATE TABLE IF NOT EXISTS ‘Spese_Preventivate‘ (
‘Anno‘ INT(4) NOT NULL,
‘Causale‘ VARCHAR(255) NOT NULL,
‘Importo‘ DECIMAL(10,2) NOT NULL,
‘TabellaFK‘ VARCHAR(1) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE ‘Spese_Preventivate‘
ADD PRIMARY KEY (‘Anno‘,‘Causale‘), ADD KEY ‘fk_Spese_Preventivate_Tabella_Millesima
(‘TabellaFK‘), ADD KEY ‘Anno‘ (‘Anno‘) USING BTREE;

Tabella **Tabelle_Millesimali**

DROP TABLE IF EXISTS ‘Tabella_Millesimale‘;
CREATE TABLE IF NOT EXISTS ‘Tabella_Millesimale‘ (
‘Condomino‘ VARCHAR(5) NOT NULL,
‘Tabella‘ VARCHAR(1) NOT NULL,
‘Millesimi‘ DECIMAL(6,2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE ‘Tabella_Millesimale‘
ADD PRIMARY KEY (‘Tabella‘,‘Condomino‘), ADD KEY ‘fk_Tabella_Millesimale_Condomini1
(‘Condomino‘), ADD KEY ‘Tabelle‘ (‘Tabella‘) USING BTREE;

Di seguito sono, poi, riportate le viste create per facilitare l’accesso e la lettura dei dati:

CREATE VIEW ‘Effettivapercond‘ AS SELECT ‘C‘.‘Interno_PI‘ AS ‘Interno_PI‘,‘C‘.‘Nome‘ AS ‘Nome‘,‘C‘.‘Cognome‘ AS ‘Cognome‘,((‘spe‘.‘Importo‘ * ‘tab‘.‘Millesimi‘) / 1000) AS ‘SpesaperCond‘,‘spe‘.‘Anno‘ AS ‘Anno‘,MONTH(‘spe‘.‘Data‘) AS ‘Mese‘,‘spe‘.‘Causale‘ AS ‘Causale‘,‘spe‘.‘Ord_Straord‘ AS ‘Ord_Straord‘ FROM ((‘Spese_Effettive‘ ‘spe‘ JOIN ‘Tabella_Millesimale‘ ‘tab‘ ON((‘spe‘.‘TabellaFK‘ = ‘tab‘.‘Tabella‘))) JOIN ‘Condomini‘ ‘C‘ ON((‘C‘.‘Interno_PI‘ = ‘tab‘.‘Condomino‘)));
La vista mostra le spese effettive per singolo condomino.

CREATE VIEW ‘OrdinarieAnno‘ AS SELECT DATABASE() AS ‘NomeCondominio‘,‘M‘.‘DescrizioneMese‘ AS ‘mesi‘,‘Q‘.‘Anno‘ AS ‘Anno‘,‘Q‘.‘Interno_PI‘ AS ‘Interno_PI‘,‘C‘.‘Nome‘ AS ‘Nome‘,‘C‘.‘Cognome‘ AS ‘Cognome‘,IF((‘Q‘.‘Interno_PI‘ LIKE ’%_I’),’Inquilino’,’Proprietario’) AS ‘PI‘,‘Q‘.‘Note‘ AS ‘Note‘,‘Q‘.‘Importo‘ AS ‘Importo‘ FROM ((‘Mesi‘ ‘M‘ JOIN ‘Quote‘ ‘Q‘ ON((‘Q‘.‘Mese‘ = ‘M‘.‘CodMese‘))) JOIN ‘Condomini‘ ‘C‘ ON((‘Q‘.‘Interno_PI‘ = ‘C‘.‘Interno_PI‘))) WHERE ((‘Q‘.‘Anno‘ = (SELECT ‘Anni‘.‘Anno‘ FROM ‘Anni‘ WHERE (‘An-

ni‘.‘InCorso‘ = 1))) AND (‘Q‘.‘OrdStraord‘ = 0));
La vista mostra le spese ordinarie annue di ogni condomino.

CREATE VIEW ‘QuoteCondomini‘ AS SELECT ‘M‘.‘DescrizioneMese‘
AS ‘Mese‘,‘Q‘.‘Anno‘ AS ‘Anno‘,IF(‘Q‘.‘Importo‘,’No’,’SI’) AS ‘Pagato‘,concat(‘Q‘.‘Importo‘,’&euro
AS ‘Importo‘,‘Q‘.‘Note‘ AS ‘Note‘,‘Q‘.‘Interno_PI‘ AS ‘Interno_PI‘,‘Q‘.‘Data_Pagamento‘
AS ‘Data_Pagamento‘,‘Q‘.‘Modo_Pagamento‘ AS ‘Modo_Pagamento‘,IF(‘Q‘.‘OrdStraord‘,’STR’,
AS ‘Gestione‘ FROM (‘Quote‘ ‘Q‘ JOIN ‘Mesi‘ ‘M‘ ON((‘Q‘.‘Mese‘ = ‘M‘.‘CodMese‘)));
Mostra le quote mensili che ogni condomino deve pagare.

CREATE VIEW ‘SpesaAnnoxCond‘ AS SELECT ‘C‘.‘Interno_PI‘ AS
‘Interno_PI‘,‘C‘.‘Nome‘ AS ‘Nome‘,‘C‘.‘Cognome‘ AS ‘Cognome‘,SUM(((‘spe‘.‘Importo‘
* ‘tab‘.‘Millesimi‘) / 1000)) AS ‘SpesaAnnualeCond‘,‘spe‘.‘Anno‘ AS ‘An-
no‘ FROM ((‘Spese_Preventivate‘ ‘spe‘ JOIN ‘Tabella_Millesimale‘ ‘tab‘
ON((‘spe‘.‘TabellaFK‘ = ‘tab‘.‘Tabella‘))) JOIN ‘Condomini‘ ‘C‘ ON((‘C‘.‘Interno_PI‘
= ‘tab‘.‘Condomino‘))) WHERE (‘spe‘.‘Anno‘ = (SELECT ‘Anni‘.‘Anno‘
FROM ‘Anni‘ WHERE (‘Anni‘.‘InCorso‘ = 1))) GROUP BY ‘C‘.‘Interno_PI‘,‘C‘.‘Nome‘,‘C‘.‘Cogno
La vista indica l’ammontare delle spese annue per ogni condomino.

CREATE VIEW ‘Straordinariaxmese‘ AS SELECT ‘Effettivapercond‘.‘Interno_PI‘
AS ‘Interno_Pi‘,SUM(‘Effettivapercond‘.‘SpesaperCond‘) AS ‘SpesaperCond‘,‘Effettivapercond‘.‘A
AS ‘Anno‘,‘Effettivapercond‘.‘Mese‘ AS ‘Mese‘,group_concat(‘Effettivapercond‘.‘Causale‘
separator ’ ’) AS ‘Causali‘ FROM ‘Effettivapercond‘ WHERE ((‘Effettiva-
percond‘.‘Ord_Straord‘ = 1) AND (‘Effettivapercond‘.‘Anno‘ = (SELECT
‘A‘.‘Anno‘ FROM ‘Anni‘ ‘A‘ WHERE (‘A‘.‘InCorso‘ = 1)))) GROUP BY ‘Ef-
fettivapercond‘.‘Interno_PI‘,‘Effettivapercond‘.‘Mese‘,‘Effettivapercond‘.‘Anno‘;
La vista mostra le spese straordinarie mensili di ogni condomino.

CREATE VIEW ‘StraordinarieAnno‘ AS SELECT DATABASE() AS
‘NomeCondominio‘,‘M‘.‘DescrizioneMese‘ AS ‘mesi‘,‘Q‘.‘Anno‘ AS ‘Anno‘,‘Q‘.‘Interno_PI‘
AS ‘Interno_PI‘,‘C‘.‘Nome‘ AS ‘Nome‘,‘C‘.‘Cognome‘ AS ‘Cognome‘,IF((‘Q‘.‘Interno_PI‘
LIKE ’%_I’),’Inquilino’,’Proprietario’) AS ‘PI‘,‘Q‘.‘Note‘ AS ‘Note‘,‘Q‘.‘Importo‘
AS ‘Importo‘ FROM ((‘Mesi‘ ‘M‘ JOIN ‘Quote‘ ‘Q‘ ON((‘Q‘.‘Mese‘ = ‘M‘.‘CodMese‘)))
JOIN ‘Condomini‘ ‘C‘ ON((‘Q‘.‘Interno_PI‘ = ‘C‘.‘Interno_PI‘))) WHE-
RE ((‘Q‘.‘Anno‘ = (SELECT ‘Anni‘.‘Anno‘ FROM ‘Anni‘ WHERE (‘An-
ni‘.‘InCorso‘ = 1))) AND (‘Q‘.‘OrdStraord‘ = 1));
La vista mostra le spese straordinarie annue straordinarie per ogni condomi-
no.

Il popolamento della base di dati
Il popolamento della base di dati avviene tramite gli statement SQL di insert. Esempio:
INSERT INTO ‘Condomini‘ (‘Interno_PI‘, ‘Nome‘, ‘Cognome‘, ‘Telefono‘, ‘Scala‘) VALUES (’a_I’, ’Mario’, ’Rossi’, 12, ’h’).

PROGETTAZIONE DELLE APPLICAZIONI

A.3.1 Livello dati

Riportiamo di seguito i trigger e le procedure atti al corretto funzionamento della base di dati:

```
CREATE TRIGGER ‘Auto Increment‘
BEFORE INSERT ON ‘Anni‘
FOR EACH ROW
SET NEW.anno = (SELECT MAX(anno)+1 FROM Anni)
```

Il seguente trigger incrementa automaticamente il valore massimo tra gli anni.

```
CREATE TRIGGER ‘LockD‘
BEFORE DELETE ON ‘Mesi‘
FOR EACH ROW
SIGNAL SQLSTATE ’45000’ SET MESSAGE_TEXT = ’An error occurred.
Table is Read-only’, MYSQL_ERRNO = 1001
CREATE TRIGGER ‘LockI‘
BEFORE INSERT ON ‘Mesi‘
FOR EACH ROW
SIGNAL SQLSTATE ’45000’SET MESSAGE_TEXT = ’An error occurred.
Table is Read-only’, MYSQL_ERRNO = 1001
CREATE TRIGGER ‘LockU‘
BEFORE UPDATE ON ‘Mesi‘
FOR EACH ROW
SIGNAL SQLSTATE ’45000’SET MESSAGE_TEXT = ’An error occurred.
Table is Read-only’, MYSQL_ERRNO = 1001
```

I tre trigger sopracitati impediscono l’Insert, l’Update e la Delete sulla tabella *Mesi*, in modo da renderla un tabella di sola lettura.

```
CREATE TRIGGER ‘AggiornaStraordinarie‘
```

```
AFTER UPDATE ON ‘Spese_Effettive‘
FOR EACH ROW
BEGIN
IF MONTH(OLD.DATA)!=MONTH(NEW.DATA) THEN
DELETE FROM ‘Quote‘ WHERE‘Quote‘.‘mese‘=MONTH(OLD.DATA);
END IF;
CALL ‘AggiornaStraord‘();
END

CREATE TRIGGER ‘CancellaStraordinarie‘
AFTER DELETE ON ‘Spese_Effettive‘
FOR EACH ROW
BEGIN
DELETE FROM ‘Quote‘ WHERE‘Quote‘.‘mese‘=MONTH(OLD.DATA);
CALL ‘AggiornaStraord‘();
END

CREATE TRIGGER ‘InserisciStraordinarie‘
AFTER INSERT ON ‘Spese_Effettive‘
FOR EACH ROW
CALL ‘AggiornaStraord‘()
```

I tre trigger sopra elencati dopo ogni Insert, Update o Delete sulle *Spese Effettive* aggiorna le *Spese Straordinarie* tramite la procedura *AggiornaStraord()*.

```
CREATE TRIGGER ‘Aggiorna Quote Delete‘
AFTER DELETE ON ‘Spese_Preventivate‘
FOR EACH STATEMENT
CALL AggiornaQuote()

CREATE TRIGGER ‘Aggiorna Quote Insert‘
AFTER INSERT ON ‘Spese_Preventivate‘
FOR EACH STATEMENT
CALL AggiornaQuote()

CREATE TRIGGER ‘Aggiorna Quote Update‘
AFTER UPDATE ON ‘Spese_Preventivate‘
FOR EACH STATEMENT
```

```
CALL AggiornaQuote()
```

I tre trigger sovraelencati dopo una Insert, Update e Delete sulla tabella *Spese Preventivate* aggiornano la tabella *Quote* tramite la procedura *AggiornaQuote()*.

```
CREATE PROCEDURE ‘AggiornaQuote‘()
BEGIN
DECLARE i INT;
SET i=1;
WHILE i < 13 DO

INSERT INTO Quote
(Mese,Anno,Importo,Interno_PI,OrdStraord)
SELECT
i,S.Anno,S.SpesaAnnualeCond/12,S.Interno_PI
FROM SpesaAnnoxCond S
ON DUPLICATE KEY UPDATE Importo=(SELECT SpesaAnnualeCond/12
FROM
SpesaAnnoxCond
WHERE
(Quote.Interno_PI=SpesaAnnoxCond.Interno_PI)
&& (Quote.Anno=SpesaAnnoxCond.Anno));
SET i = i + 1;
END WHILE ;
END
```

La procedura sovrastante aggiorna le quote ordinarie prendendo i dati dalla vista *Spesaannoxcond*

```
CREATE PROCEDURE ‘AggiornaStraord‘()
NO SQL
INSERT INTO Quote (Mese,Anno,Importo,Note,Interno_PI,OrdStraord)
SELECT S.Mese,S.Anno,S.SpesaperCond,S.Causali,S.Interno_PI
FROM ‘Straordinariaxmese‘ S
ON DUPLICATE KEY UPDATE Importo=S.SpesaperCond,Note=.S.Causali
END
```

La procedura aggiorna le quote straordinarie prendendo i dati dalla vista *Straordinariexmese*

```
CREATE PROCEDURE ‘CreateUser‘(IN ‘Username‘ VARCHAR(20), IN
‘Pass‘ VARCHAR(20))
NO SQL
BEGIN
SET @usr=Username;
SET @pwd=Pass;
SET @query1 = CONCAT(’
CREATE USER ’,@usr,’@% IDENTIFIED BY ’,@pwd,’ ’);
PREPARE stmt FROM @query1;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;
SET @query1 = CONCAT(’
GRANT SELECT ON ‘Condominio_test‘.‘QuoteCondomini‘ TO ’,@usr,’@%
WITH MAX_QUERIES_PER_HOUR 100 MAX_CONNECTIONS_PER_HOUR
60
MAX_USER_CONNECTIONS 10’
);
PREPARE stmt FROM @query1; EXECUTE stmt;
DEALLOCATE PREPARE stmt;
END

CREATE PROCEDURE ‘DeleteUser‘(IN ‘UserName‘ VARCHAR(20))
NO SQL
BEGIN
SET @usr=UserName;
SET @query1 = CONCAT(’
DROP USER ’,@usr,’@%’);
PREPARE stmt FROM @query1;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;
END
```

Le due procedure sopracitate creano e cancellano gli utenti di tipo condomino della base di dati.

A.3.2 Il livello applicazione

Sono state inoltre progettate due applicazioni:
la prima su sistema operativo Android per permettere al condomino di visualizzare le proprie quote. Questa è stata implementata attraverso l'ADT e un backend in php che provvede alla connessione al database e fornisce un interfaccia web ai condomini nel caso non fossero in possesso di uno smartphone. L'Amministratore provvederà a creare una corrispondenza 1:1 tra l'interno e l'username associato al condomino in quanto l'applicazione effettua la seguente QUERY sul database:
SELECT * FROM QuoteCondomini WHERE 'Interno_PI' = USERNAME;
Il login viene effettuato tramite credenziali fornite al singolo condomino, poi viene utilizzata una classe MCrypt per cifrare i dati tra l'applicazione in Java e il backend in PHP in AES.
I dati delle quote vengono presentati al condomino sotto forma tabellare.
In futuro sono previsti miglioramenti dell'APP con l'aggiunta di ulteriori funzionalità (Visualizzazione Bilancio, Segnala Problemi all'Amministratore ecc..)
L'Applicazione Desktop per l'amministratore è stata sviluppata, invece, con la piattaforma Visual Studio attraverso l'uso del .NET Framework e del connettore per MYSQL per il suddetto.
L'Applicazione è stata creata con una struttura modulare attraverso l'uso del paradigma ad Oggetti, divisa in varie librerie dinamiche (dll), che le permettono di avere una maggiore velocità di esecuzione a run-time e una maggiore manutenibilità software.
• Modulo Principale
Provvede al login dell'Amministratore tramite le credenziali del database e al caricamento dell'interfaccia principale.
Le credenziali vengono passate al modulo, Autenticazione il quale provvede a normalizzarle con un'espressione regolare del tipo [^ a-zA-Z0-9], " " che sostituisce a tutti i caratteri non alfanumerici spazi bianchi in modo da prevenire eventuali Injection. Inoltre, il modulo le rende disponibili globalmente a tutti gli altri moduli del programma.
L'Interfaccia presenta l'accesso alle varie funzioni del programma attraverso una struttura semplice ed essenziale.

• Modulo Opzioni
Nel Menù Condomino ritroviamo le opzioni per modificare l'annualità, inserirne una nuova, modificare i modi di pagamento e per verificare la consistenza delle tabelle millesimali (Somma Millesimi=1000).
• Modulo Anagrafica
Nel Menù Anagrafica ritroviamo invece le opzioni per l'inserimento la modifica e la gestione dei condomini e dei relativi millesimi associati. Si è provveduto a creare anche un layout form oltre al semplice editor in tabella per la gestione dei condomini per una maggiore operatività.
• Modulo Quote
Nel Menù Quote troviamo il centro del nostro Gestionale Condomini, infatti qui è possibile vedere e modificare le quote relative ai condomini automaticamente generate dal database e creare il file per la stampa delle quietanze di pagamento relative alle varie mensilità.
•Modulo Spese
Nel Menù Spese, infine, troviamo la gestione delle spese preventive ed effettive con l'usuale possibilità di inserimento, aggiornamento e cancellazione.
Nell'applicazione deve essere ancora implementata la gestione del bilancio finale del condominio, per la quale sarà richiesta un'ulteriore analisi dei requisiti dovuta alla complessa natura fiscale e finanziaria del problema. In questo modo, produrremo, per quanto possibile, un servizio più General-Purpose.
Di seguito si riportano la mappa delle dipendenze e le varie mappe classi.

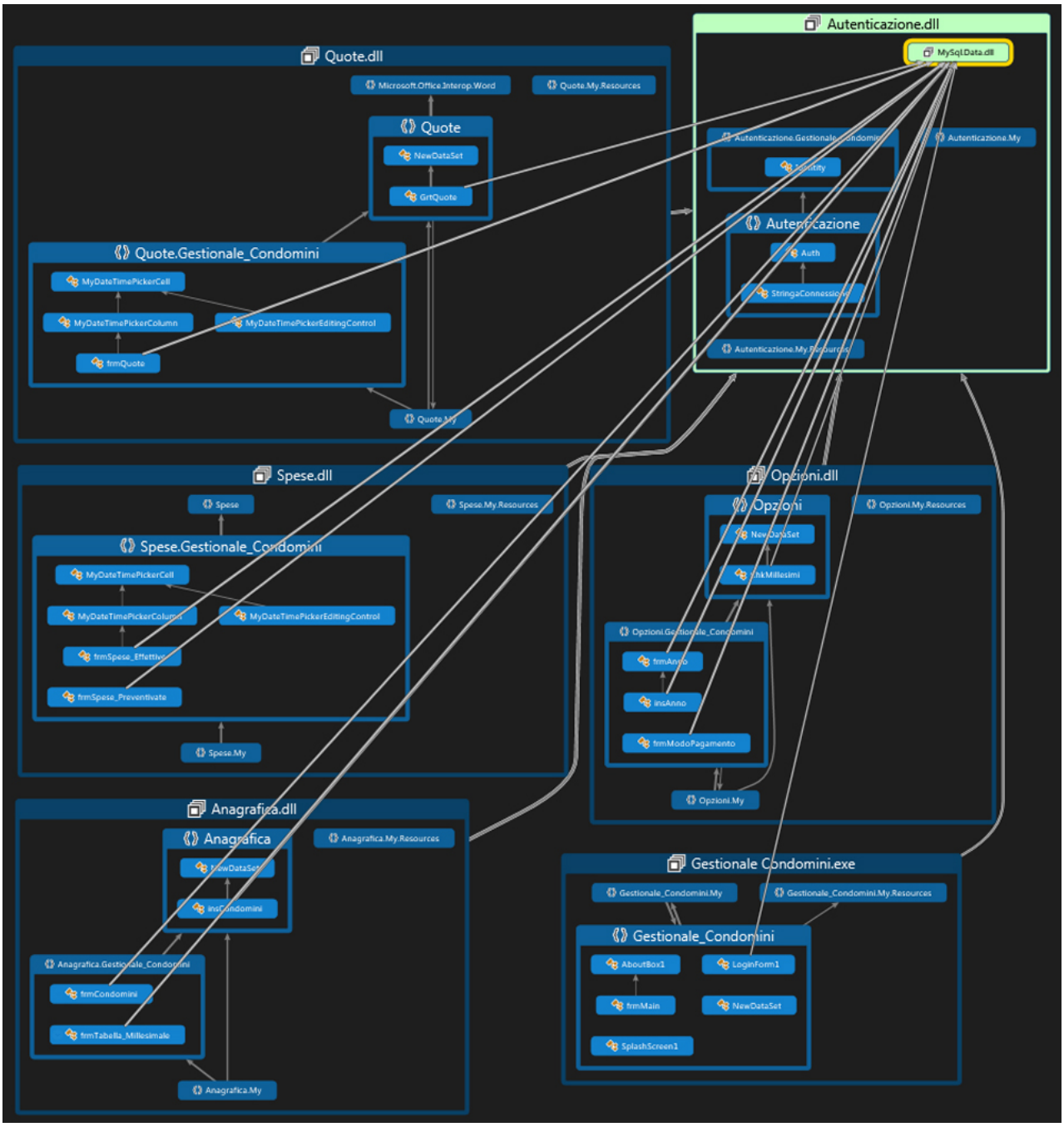

Figura 4: Mappa dei codici

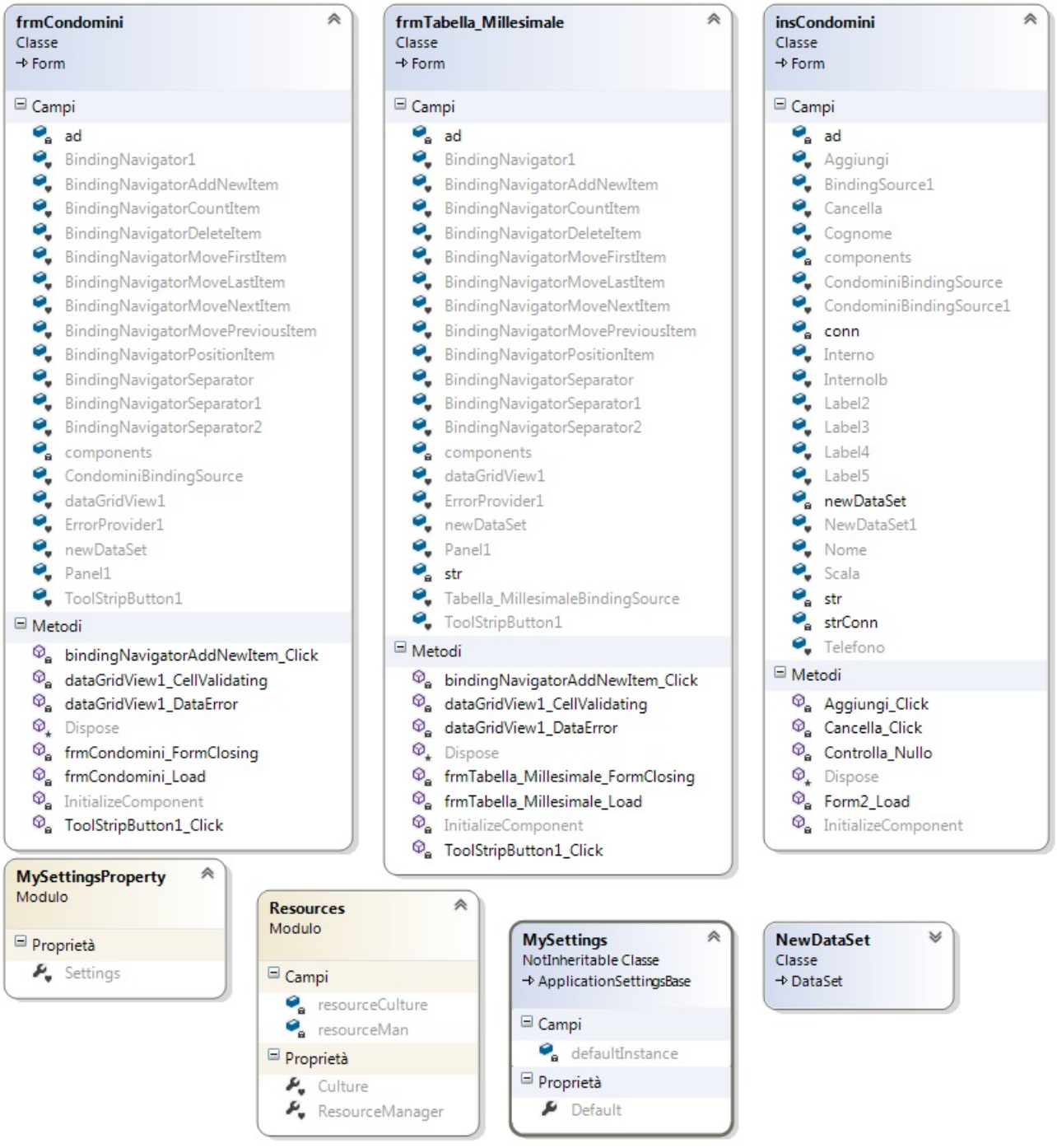

Figura 5: Diagramma delle classi: Anagrafica

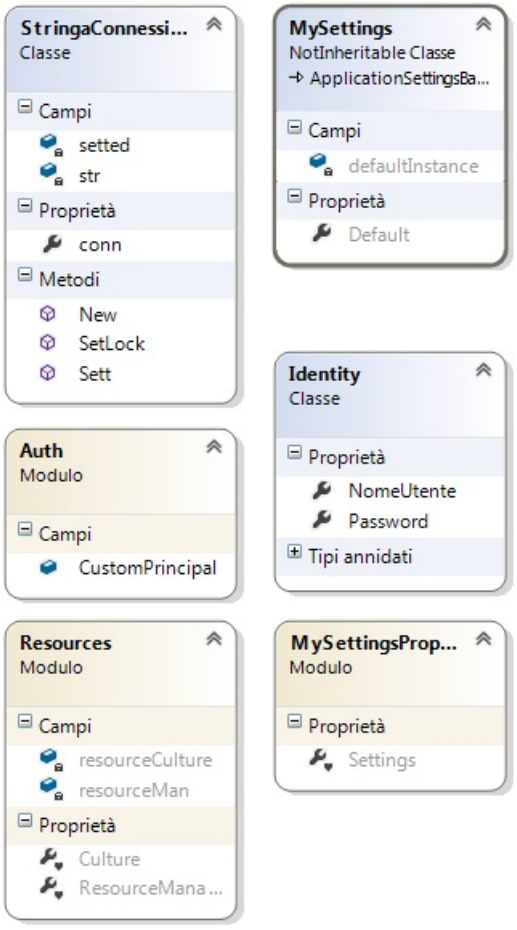

Figura 6: Diagramma delle classi: Autenticazione

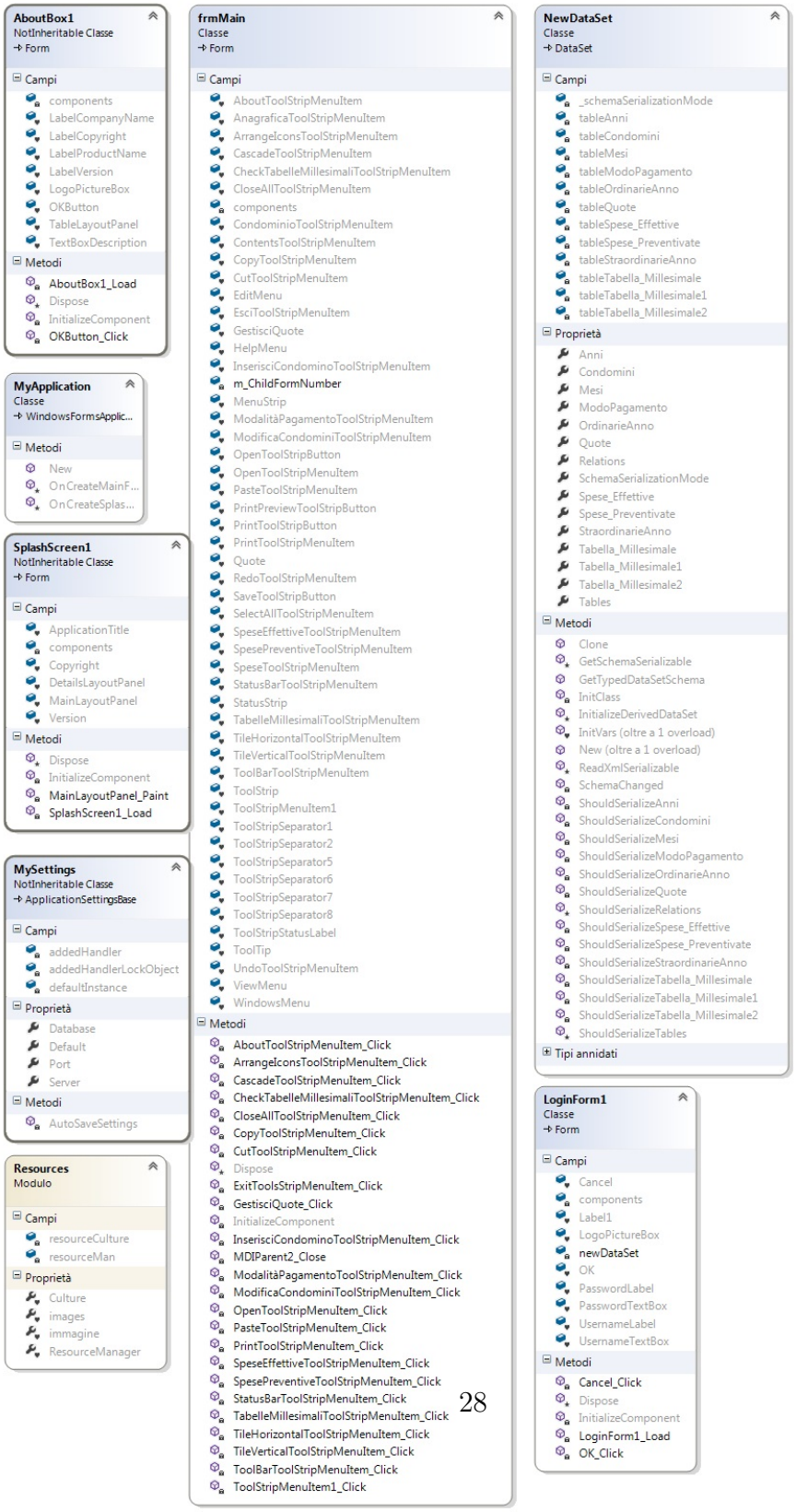

Figura 7: Diagramma delle classi: Main

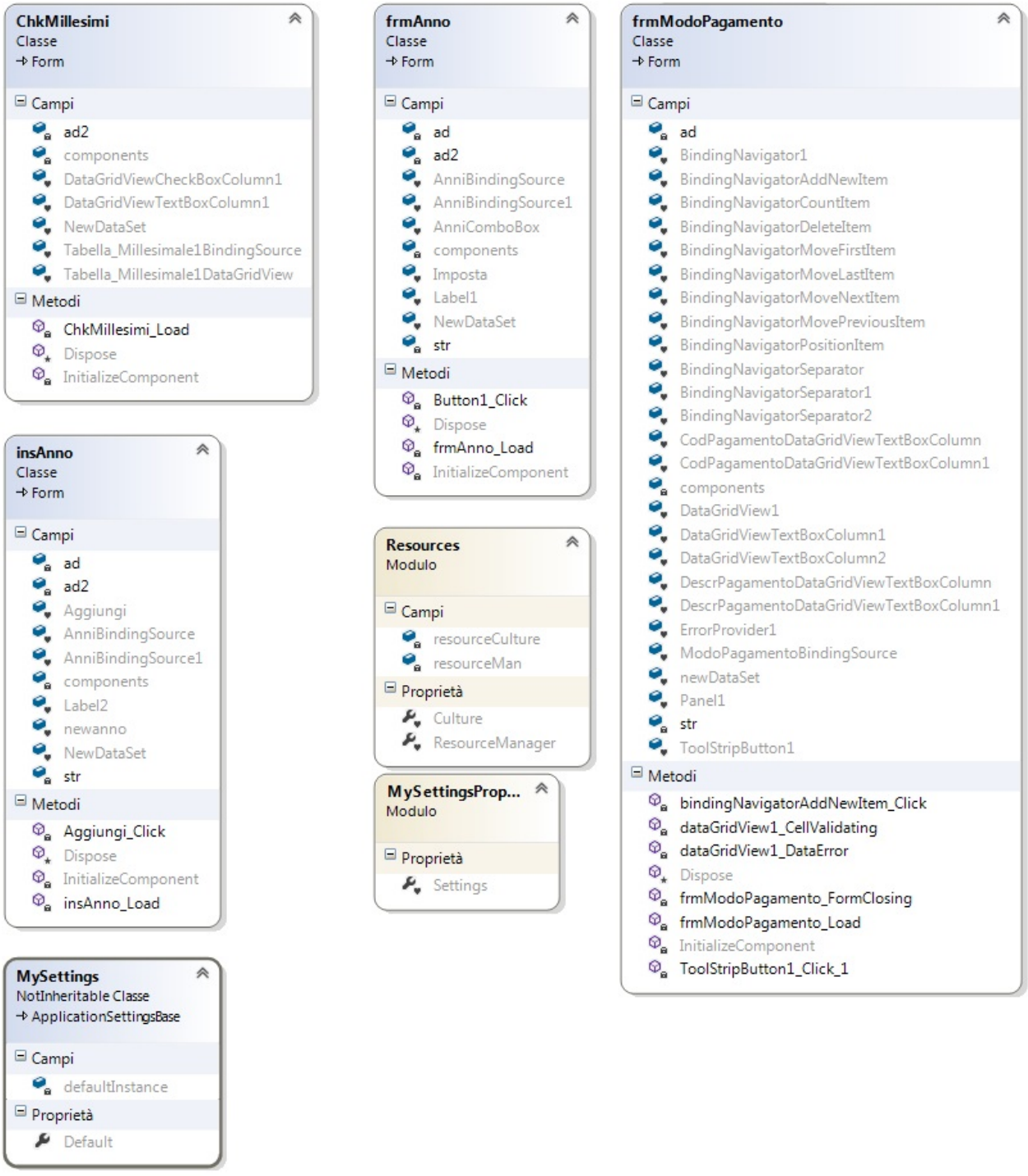

Figura 8: Diagramma delle classi: Opzioni

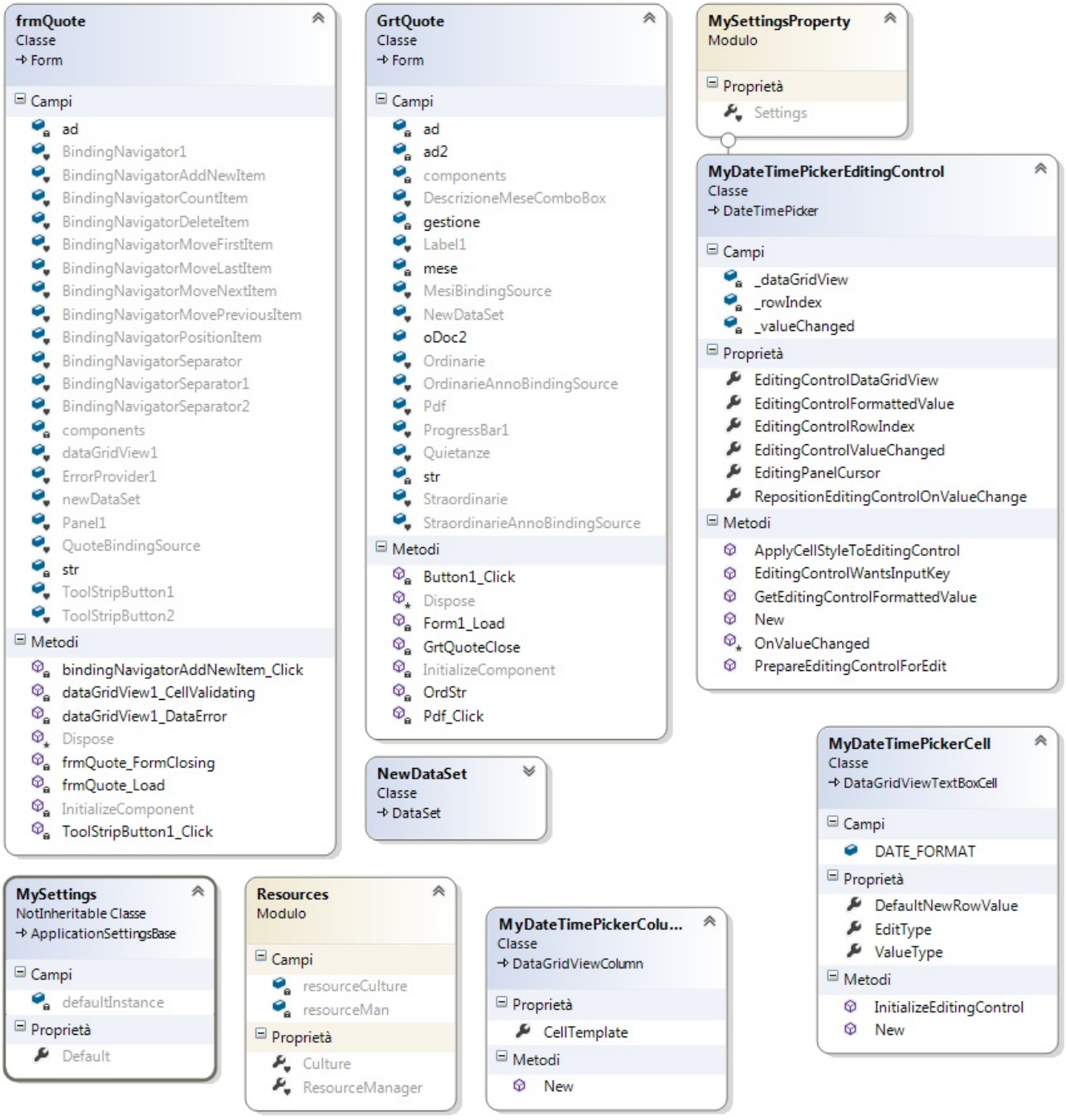

Figura 9: Diagramma delle classi: Quote

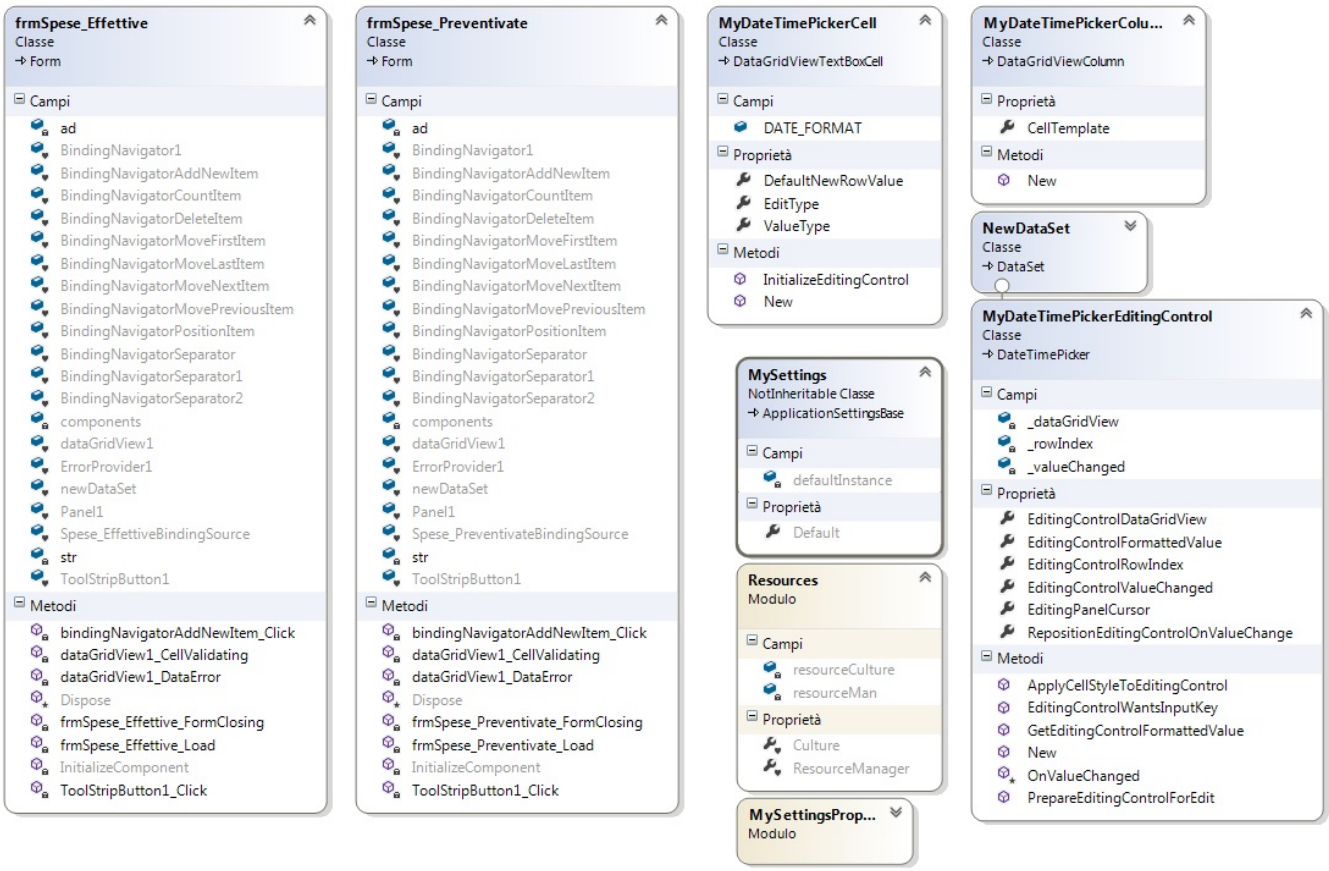

Figura 10: Diagramma delle classi: Spese

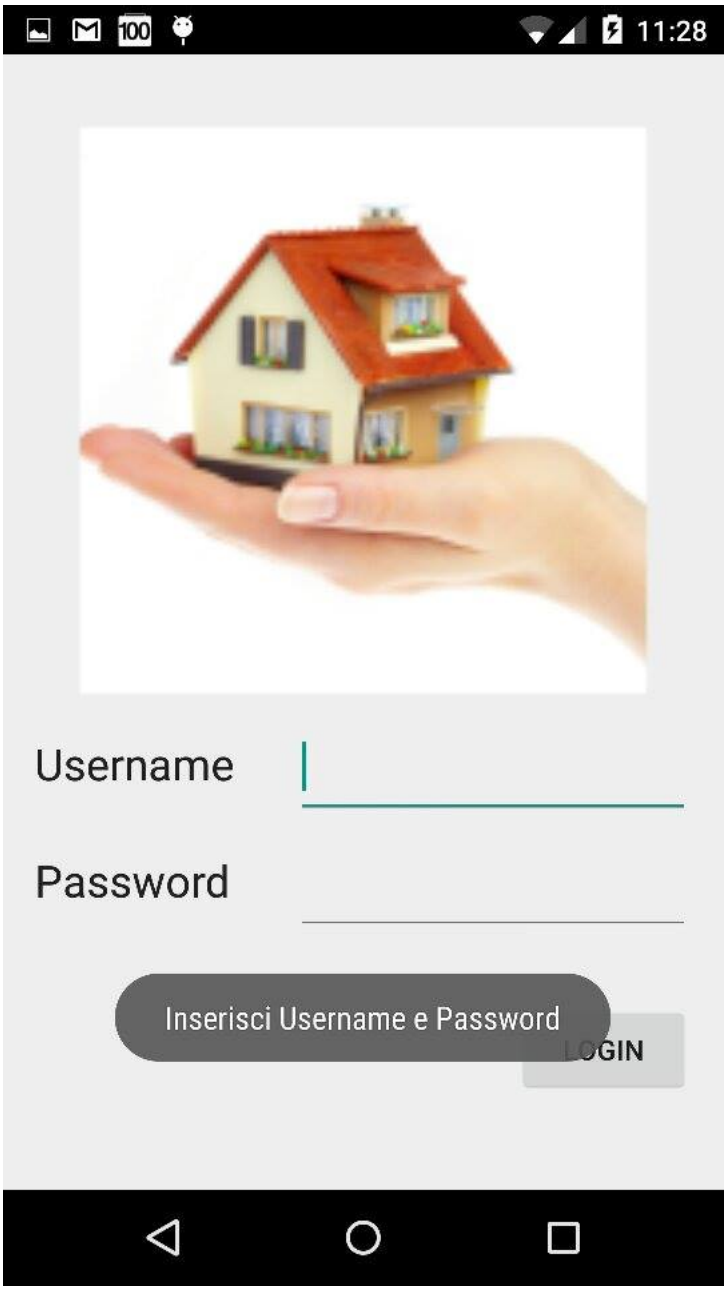

Figura 11: Applicazione Android: Login

Figura 12: Applicazione Android: Tabella Quote