Università degli studi di Trieste

PROGETTAZIONE DI UNA BASE DI DATI PER IL PIPELINE PIGGING

Giovanni Coronica

Trieste, 05/06/2022

Sommario

Analisi dei requisiti **3**
Requisiti espressi in linguaggio naturale 3
Glossario dei termini 4
Ristrutturazione dei requisiti 5
Progettazione concettuale **6**
Identificazione delle entità e relazioni 6
Analisi delle entità e delle relazioni 8
Schema concettuale 11
Vincoli non esprimibili 11
Progettazione logica **12**
Ristrutturazione schema E-R 12
Tavola dei volumi 12
Valutazione dei costi 12
Analisi delle ridondanze 13
Eliminazione delle generalizzazioni 14
Scelta degli identificatori primari 15
Schema E-R finale 16
Schema logico 17
Interazioni con il DB e programmabilità **19**
Operazioni sul Database (Query) 19
Codice Java 20
Vincoli e Triggers 21

Analisi dei requisiti

Requisiti espressi in linguaggio naturale

Si vuole realizzare un Data Base sul Pipeline Pigging, ovvero un database che tenga conto delle corse di uno strumento chiamato Pig lungo un tubo di un oleodotto petrolifero, in modo tale da pulire la condotta o rilevare eventuali anomalie nella parete del tubo. Si vogliono principalmente memorizzare i dati della singola corsa del Pig, con quale strumento è stata effettuata e quali anomalie sono state rilevate durante l'operazione;

Importante per la Corsa del Pig è la data e l'ora di partenza, la data e l'ora di arrivo del Pig ed eventuali Interruzioni della corsa; ovviamente quando il Pig arriva a destinazione deve essere sottoposto a delle pulizie, per essere poi rimandato indietro senza essere reinserito nella tubatura (in quanto questa ha un solo senso di pompaggio); quindi il Pig non può essere riutilizzato prima di un tempo minimo di una settimana; non possono inoltre esistere due corse contemporaneamente: prima di iniziare una nuova corsa, deve essersi conclusa la precedente; lo scopo dell'operazione è quello di pulire il tubo o di rilevare eventuali anomalie.

Durante la corsa possono essere rilevate delle Anomalie le quali possono avere un diverso livello di gravità; ci interessa sapere in quale sezione del tubo è presente quest'anomalia, il kilometro (con la precisione del metro) lungo la linea al quale essa si presenta e sapere se, nel caso in cui l'anomalia fosse vecchia, è stata già effettuata una manutenzione; si vuole poi salvare una breve descrizione dell'anomalia e, se presenti, delle scansioni effettuate dal Pig; inoltre certi Pig sono in grado anche di misurare la temperatura della parete del tubo quando hanno rilevato la corrosione.

Quando viene rilevata un'anomalia, bisogna assolutamente sapere in quale Sezione di tubo questa è presente; per questa si vuole sapere: di quale materiale è composta, quali sono le sue dimensioni (diametro e lunghezza), qual è la data della sua ultima manutenzione, la data di installazione e quante corrosioni presenta; ovviamente se la data della sua ultima manutenzione precede la data della manutenzione di una sua anomalia, allora questa deve essere aggiornata alla data di manutenzione dell'anomalia.

Per quanto riguarda la Fabbrica della sezione, vogliamo memorizzare la partita IVA, forma giuridica, ragione sociale, sede legale (indirizzo, comune e provincia), il numero di telefono ed eventualmente il sito web.

Per quanto riguarda il Pig, esso può essere di proprietà della società dell'oleodotto o può essere stato fornito da una ditta esterna; si vuole memorizzare nel caso di acquisto, la data nella quale questo è avvenuto; vogliamo poi principalmente sapere il numero di corse che ha effettuato e le sue dimensioni (diametro e lunghezza); il Pig può essere di più tipi: Utility Pig cioè uno strumento di pulitura dotato di spazzole o uno Smart Pig che fornisce i dati sulla situazione delle pareti del tubo; solamente lo Smart Pig rileva le anomalie.

Se il Pig è di fornitura esterna, vogliamo memorizzare i dati della Ditta fornitrice quali partita IVA, forma giuridica, ragione sociale, sede legale dotata di indirizzo, comune e provincia, il numero di telefono ed eventualmente il sito web; vogliamo inoltre tenere conto di quante volte questa ditta ha fornito Pig esterni.

In fine si vogliono gestire eventuali Interruzioni della corsa del Pig, memorizzando il kilometro al quale è avvenuta e una breve descrizione del perché essa è avvenuta; ovviamente se una corsa viene interrotta la si segna come conclusa.

Glossario dei termini

TERMINE	DESCRIZIONE	SINONIMO	COLLEGAMENTO
Sezione	La sezione singola che compone il tubo dell'oleodotto.	Parte di tubo	Anomalia, Fabbrica
Anomalia	Le anomalie che sono state rilevate dalla corsa del Pig.	Corrosione	Corsa Pig, Sezione
Corsa del Pig	La corsa del pig lungo il tubo dell'oleodotto.	Corsa, operazione	Pig, Anomalie
Pig	Lo strumento di rilevazione delle anomalie che viene fatto correre lungo l'oleodotto; può essere un di tipo Utility o Smart.	Strumento	Corsa del Pig, Ditta
Ditta Pig	La ditta che ha fornito il pig alla società.		Pig
Interruzione	L'interruzione di una determinata corsa che non è andata a buon fine a causa di qualche problema.		Corsa del Pig
Fabbrica	La Fabbrica che ha prodotto la sezione di tubo che presenta una o più anomalie.		Sezione

Ristrutturazione dei requisiti

Frasi di carattere generale

Si vuole realizzare un Data Base sul Pipeline Pigging che tenga conto delle corse di uno strumento chiamato Pig lungo un tubo di un oleodotto petrolifero, in modo tale da pulire la condotta o rilevare eventuali anomalie nella parete del tubo.

Frasi relative alla sezione

Il tubo è diviso in **Sezioni**; per ogni singola sezione si vuole sapere di quale materiale è composta, quali sono le sue dimensioni (diametro e lunghezza), qual è la data della sua ultima manutenzione, la data di installazione e quante anomalie presenta;

Frasi relative all'anomalia

Per le **Anomalie** rilevate ci interessa sapere in quale sezione del tubo è presente quest'anomalia, il kilometro (con la precisione del metro) lungo la linea al quale essa si presenta ed inoltre il suo livello di gravità; nel caso in cui l'anomalia fosse vecchia, vogliamo sapere se è già stata effettuata una manutenzione; si vuole poi salvare una breve descrizione dell'anomalia e, se presenti, delle scansioni effettuate dal Pig; inoltre se possibile, vogliamo sapere la temperatura della parete del tubo al momento della rilevazione.

Frasi relative alla corsa del pig

Per la **Corsa del Pig** vogliamo rappresentare la data e l'ora di partenza, la data e l'ora di arrivo del Pig, eventuali Interruzioni della corsa e con quale Pig è stata effettuata; vogliamo inoltre sapere se durante l'operazione sono state rilevate delle anomalie.

Frasi relative al Pig

Per quanto riguarda il **Pig**, esso può essere di proprietà della società dell'oleodotto o può essere stato fornito da una ditta esterna; si vuole memorizzare nel caso di acquisto, la data nella quale questo è avvenuto e il prezzo; vogliamo poi principalmente sapere il numero di corse che ha effettuato e le sue dimensioni (diametro e lunghezza); il Pig può essere di più tipi: Utility Pig o Smart Pig; solamente lo Smart Pig rileva le anomalie.

Frasi relative alla ditta del pig

Se il Pig è di fornitura esterna, vogliamo memorizzare i dati della **Ditta fornitrice** quali partita IVA, forma giuridica, ragione sociale, sede legale dotata di indirizzo, comune e provincia, il numero di telefono ed eventualmente il sito web; vogliamo inoltre tenere conto di quante volte questa ditta ha prestato servizio per il pigging.

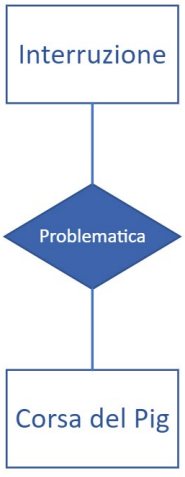
Frasi relative all'interruzione

Si vogliono gestire eventuali **interruzioni** della corsa del Pig, memorizzando il kilometro al quale è avvenuta e una breve descrizione del perché essa è avvenuta.

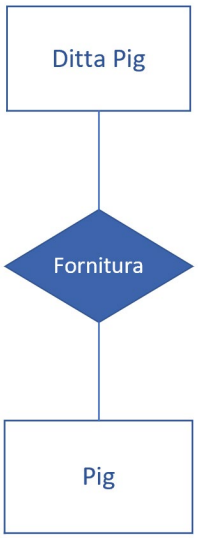
Progettazione concettuale

Identificazione entità e relazioni

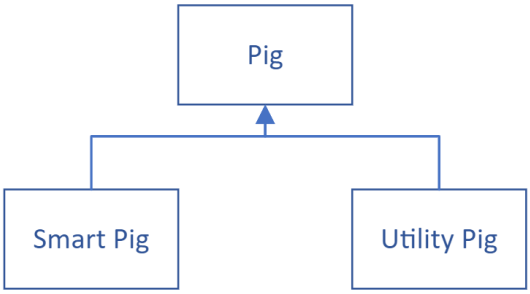
Il **Pig** viene **Utilizzato** in una **Corsa** e rileva i **Dati** delle **Anomalie**; L'anomalia **Appartiene** ad una particolare **Sezione** di tubatura; essendo questi, tutti concetti aventi proprietà significative, è stato opportuno rappresentarli con entità.

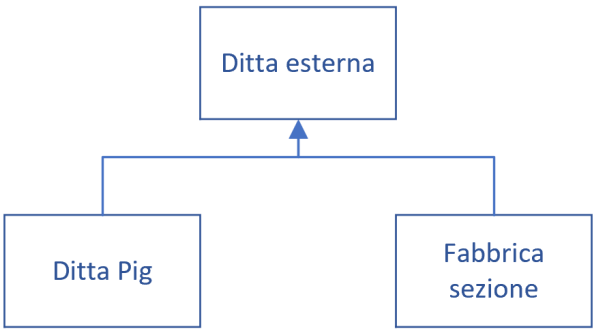


La **Corsa del Pig** può avere una e una sola **interruzione**, per la quale si vogliono avere dei dati ben precisi.



- Il **Pig** può essere di fornitura esterna e per questo si vogliono sapere i dati della **Ditta** che lo ha fornito;
- Lo strumento può essere di due tipi: **Utility** o **Smart**; questa generalizzazione è totale ed esclusiva, in quanto il Pig può deve essere o pulitore (Utility) o rilevatore (Smart);





La **Fabbrica della sezione** e la **Ditta** che fornisce il Pig sono entrambe **Ditte Esterne**; anche questa generalizzazione è totale ed esclusiva perché oltre a questi due tipi di ditte non ci sono altri casi di interesse e inoltre non può succedere che una ditta produca sia Pig che Sezioni di oleodotto.

Analisi delle entità e delle relazioni

ENTITÀ	ATTRIBUTI	DESCRIZIONE
Corsa del Pig		
	ID corsa	
	Data partenza	
	Data arrivo	In caso di interruzione rappresenta la data dell'interruzione
	Ora partenza	
	Ora arrivo	In caso di interruzione rappresenta l'ora dell'interruzione
Pig		
	ID Pig	
	Numero corse	
	Dimensioni	Composto: lunghezza (metri), diametro (centimetri)
Anomalia		
	ID anomalia	
	Progressiva	Il kilometro al quale si presenta l'anomalia
	Gravità	Numero indice della gravità dell'anomalia
	Data manutenzione	
	Scansione	
	Temperatura	
	Descrizione	
Sezione		
	ID sezione	
	Materiale	
	Dimensioni	Composto: lunghezza (metri), diametro (centimetri)
	Data ultima manutenzione	
	Data installazione	
	Numero anomalie	

Ditta Esterna

Partita IVA	
Ragione sociale	
Forma giuridica	
Sede legale	Composto: indirizzo, comune, provincia
Numero telefono	
Sito web	

Interruzione

Kilometro
Descrizione
Data
Ora

RELAZIONE	TIPO

FORNITURA	UNO A MOLTI

- Un Pig in qualunque tipologia di fornitura (acquisto o prestito) ha comunque una Ditta di fornitura e ovviamente può averne al massimo una: **(1,1)**;
- Una Ditta può aver fornito più strumenti, sia in prestito che in cessione, ma ne avrà fornito almeno uno: **(1,N)**;

Attributi:

Tipologia
Prezzo
Data acquisto

UTILIZZO	UNO A MOLTI

- Un Pig può aver partecipato a più operazioni ed almeno ad una: **(1,N)**;
- Per una Corsa può esser stato utilizzato uno e uno solo strumento, indipendentemente dalla tipologia (Utility o Smart): **(1,1)**;

APPARTENENZA	UNO A MOLTI

- Un'Anomalia può far parte di una ed una sola sezione: **(1,1)**;
- Una Sezione può presentare nessuna o più anomalie: **(0,N)**;

DATI	UNO A MOLTI

- Una Corsa può non aver rilevato anomalie per due motivi: il primo motivo lo si ha se non sono presenti anomalie nella sezione di tubatura, il secondo motivo se è stato utilizzato un Utility Pig; altrimenti la corsa può aver rilevato una o più anomalie: **(0,N)**;
- Un'anomalia può esser stata rilevata da una ed una sola corsa, in quanto appena rilevata, verrà sottoposta a manutenzione e non sarà più presente alla corsa successiva: **(1,1)**;

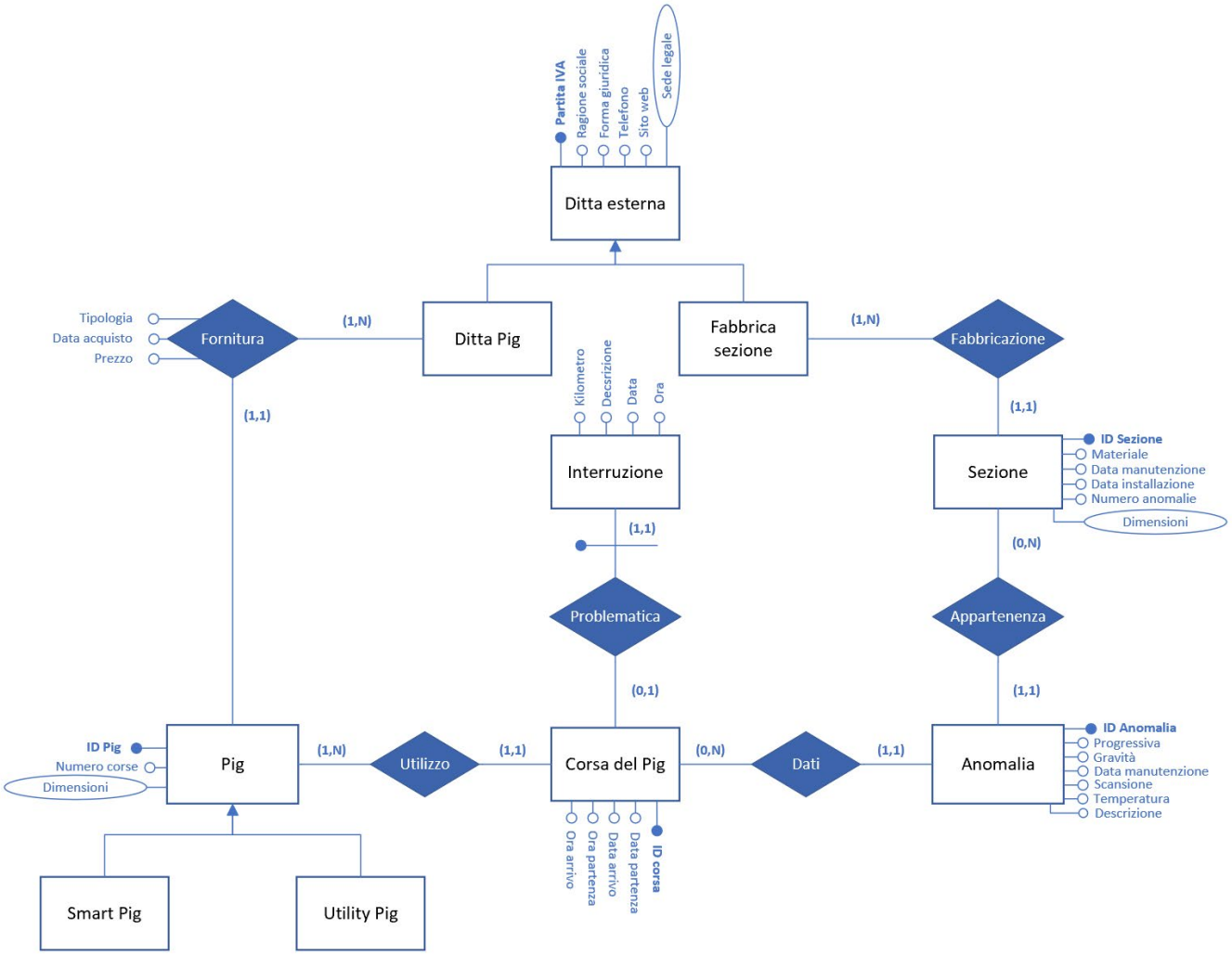
PROBLEMATICA	UNO A UNO

- Una Corsa può essere interrotta per qualche problema; se una corsa viene interrotta, viene segnata come conclusa, quindi potrà presentare o zero interruzioni o al massimo una: **(0,1)**;
- Un Interruzione, se esiste, può riferirsi ad una ed una sola corsa: **(1,1)**;

FABBRICAZIONE	UNO A MOLTI

- Una sezione che presenta un'anomalia deve essere stata fabbricata da una ed una sola fabbrica: **(1,1)**;
- Una fabbrica può aver prodotto una o più sezioni di tubatura: **(1,N)**;

Schema concettuale E-R



Vincoli non esprimibili

- La distanza temporale tra una corsa e l'altra deve essere almeno di una settimana
- L'attributo Gravità nell'entità Anomalia è un valore compreso tra 1 e 10
- La temperatura della sezione di tubatura deve essere compresa nell'intervallo di -20°C e +150°C
- Il campo Progressiva nell'entità Anomalia deve avere la precisione del metro
- Il diametro del Pig non può essere maggiore del diametro della sezione
- Solo lo Smart Pig può rilevare anomalie durante una corsa
- La data di manutenzione di un'anomalia deve essere successiva alla corsa del Pig

Progettazione logica

Ristrutturazione dello schema ER

Tavola dei volumi

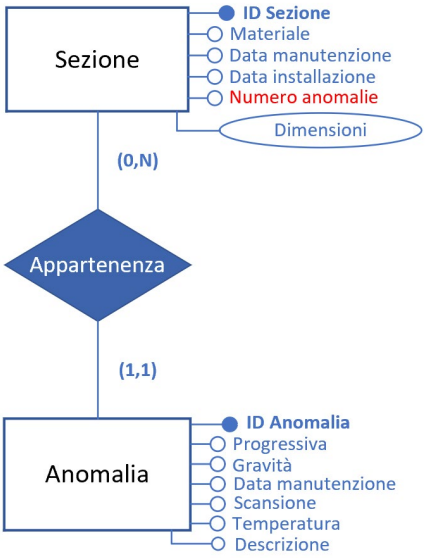
Concetto	Tipo	Volume
Anomalia	E	24000
Sezione	E	38800
Pig	E	10
Interruzioni	E	20
Fabbrica Sezione	E	10
Ditta Pig	E	5
Corsa del Pig	E	240
Fabbricazione	R	4000
Appartenenza	R	1
Dati	R	500
Utilizzo	R	10
Fornitura	R	8
Problematica	R	20

Valutazione dei costi

Operazione	Tipo	Frequenza
Aggiunta di una nuova Corsa	Interattiva	1/mese
Aggiunta dei dati di un'Anomalia	Interattiva	100/mese
Aggiunta dei dati di una Ditta Pig	Interattiva	< 1/anno
Aggiunta dei dati di una Fabbrica	Interattiva	< 1/anno
Aggiunta dei dati di una Sezione	Interattiva	10/anno
Aggiunta dei dati per un'interruzione	Interattiva	1/anno
Estrazione di tutti i dati di un'Anomalia	Interattiva	1/giorno
Estrazione di tutti i dati di una Sezione	Interattiva	5/settimana
Visualizzare il Pig che ha subito più interruzioni	Interattiva	1/anno
Data una Ditta fornitrice di un Pig, ricavare il numero di volte che ha fornito Pig esterni	Interattiva	1/mese
Dato un Pig, ricavare il numero di corse	Interattiva	1/settimana
Ricavare i dati di tutte le anomalie non ancora sottoposte a manutenzione	Batch	1/mese
Data un'Anomalia, ricavare tutti i dati della sezione di tubo	Batch	5/settimana

Analisi delle ridondanze

<u>Numero anomalie</u> **[Sezione]**



Op. 1 Aggiunta di una nuova Anomalia **100/Mese**
Op. 2 Stampa dei dati di una Sezione **5/Settimana**

CON RIDONDANZA

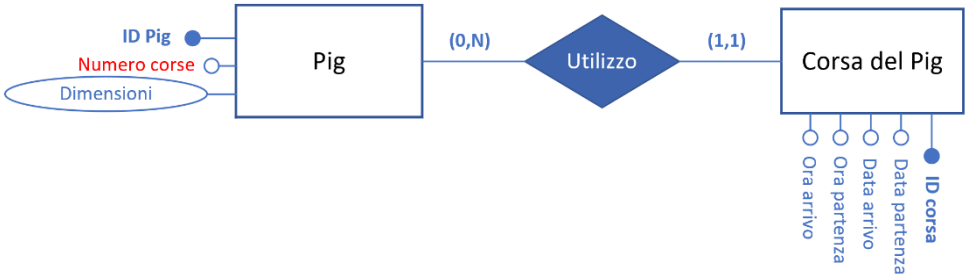
	CONCETTO	ACCESSI	TIPO
OP 1	Anomalia	1	S
	Appartenenza	1	S
	Sezione	1	S
	Sezione	1	L
OP 2	Sezione	1	L
TOTALE: **180/SETTIMANA**			

SENZA RIDONDANZA

	CONCETTO	ACCESSI	TIPO
OP 1	Anomalia	1	S
	Appartenenza	1	S
OP 2	Sezione	1	L
	Appartenenza	1	L
TOTALE: **110/SETTIMANA**			

Dalle due tabelle si evince che eliminando la ridondanza diminuiscono gli accessi settimanali. Si preferisce quindi **eliminare** il dato ridondante. (Si è supposto 1 mese = 4 settimane)

Numero corse **[Pig]**



Op. 1 Aggiunta di una nuova corsa **1/mese**
Op. 2 Dato un Pig, ricavare il numero di corse **1/mese**

CON RIDONDANZA

	CONCETTO	ACCESSI	TIPO
OP 1	Corsa del Pig	1	S
	Utilizzo	1	S
	Pig	1	L
	Pig	1	S
OP 2	Pig	1	L
TOTALE: **8/mese**			

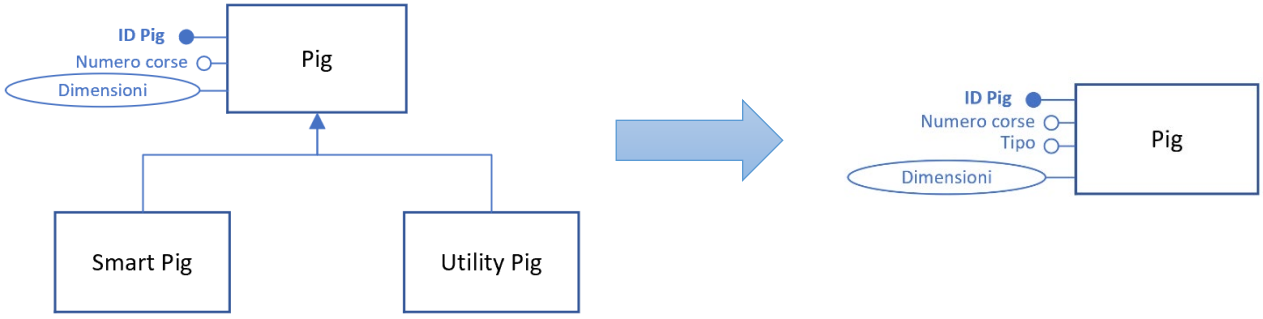
SENZA RIDONDANZA

	CONCETTO	ACCESSI	TIPO
OP 1	Corsa del Pig	1	S
	Utilizzo	1	S
OP 2	Pig	1	L
	Utilizzo	24	L
TOTALE: **29/mese**			

In presenza del dato ridondante abbiamo un totale di 8 accessi a mese, mentre in assenza del dato ridondante abbiamo 29 accessi a mese. Detto questo preferiamo **mantenere** il dato ridondante, anche se la prestazione non subirebbe drastici miglioramenti.

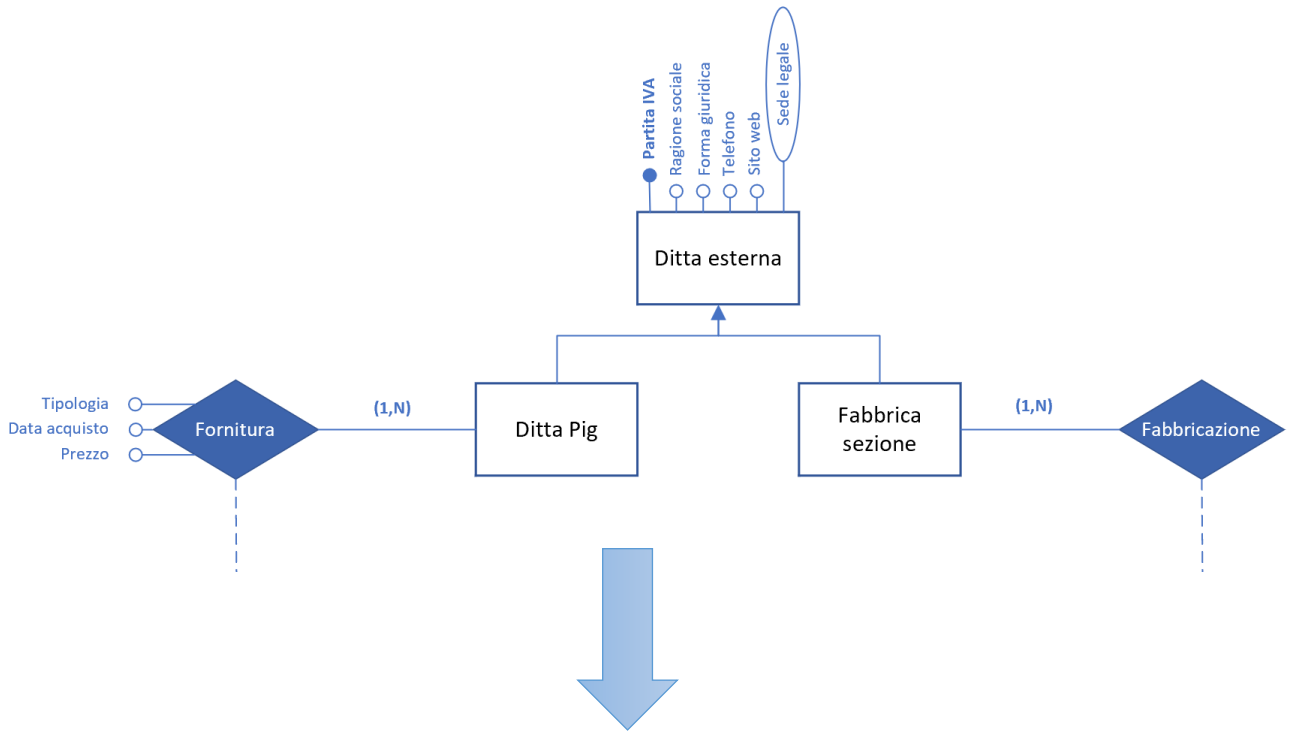
Eliminazione delle generalizzazioni

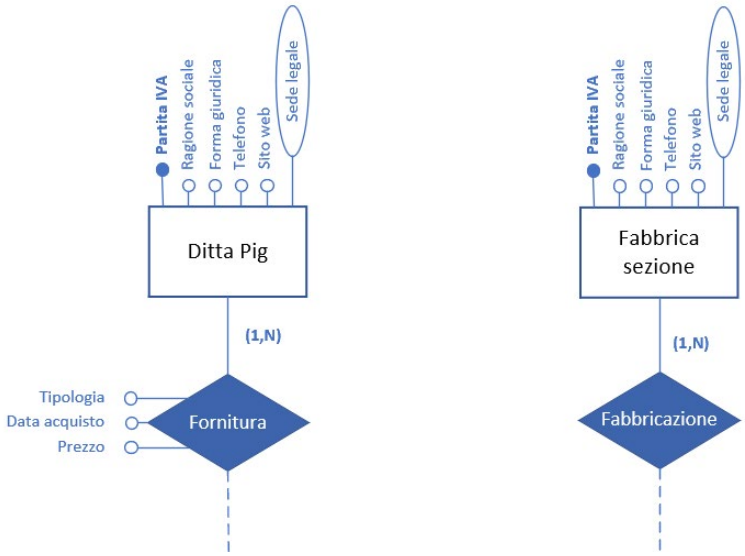
Pig	
Utility	**Smart**
Accorpamento dei figli nel genitore	



Dato che la generalizzazione è esclusiva si può eliminare questa generalizzazione accorpando i figli (Utility Pig e Smart Pig) nell'entità padre (Pig) aggiungendo un attributo **Tipo** che indentificherà il tipo di Pig (1 – Utility, 2 – Smart).

Ditta esterna	
Ditta Pig	**Fabbrica sezione**
Accorpamento del genitore nei figli	



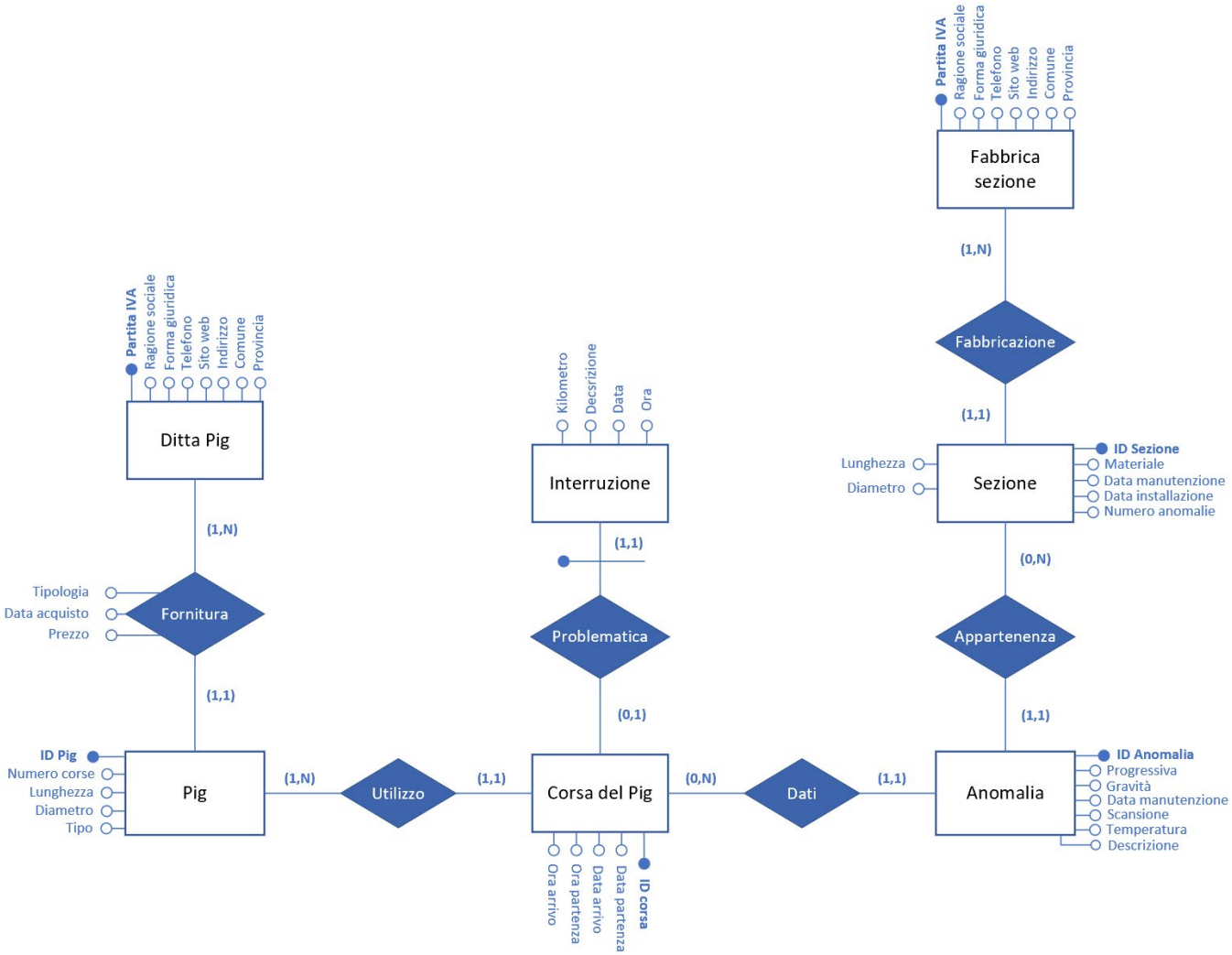


Nonostante queste entità abbiano gli stessi identici attributi si riferiscono a due concetti distinti al fine dell'accesso ai dati, e vengono utilizzate in contesti molto differenti. La soluzione scelta, quindi, è stata quella di accorpare l'entità genitore nelle entità figlie e mantenere due entità separate.

Scelta degli identificatori primari

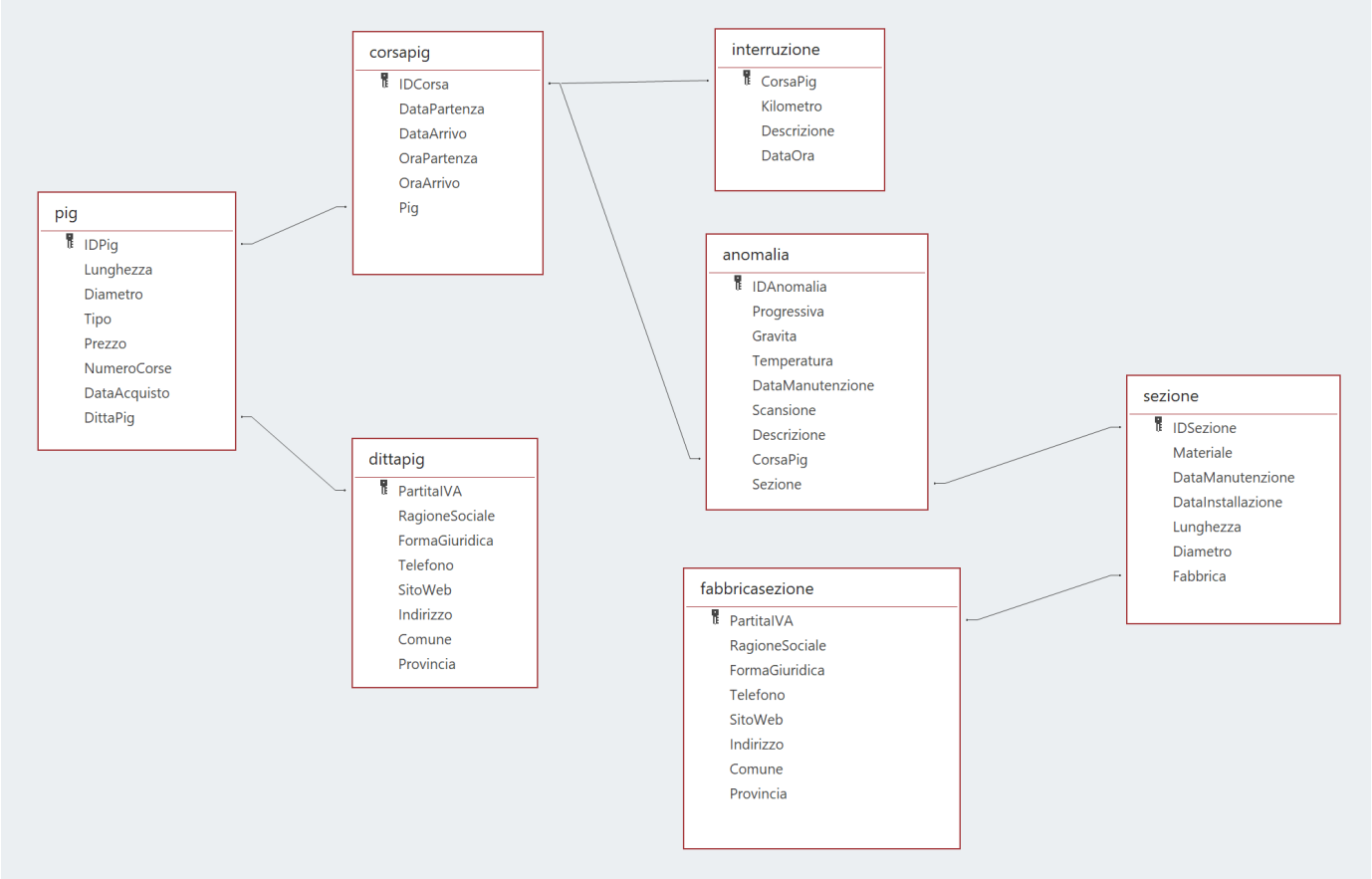
Entità	Identificatore
Corsa del pig	ID Corsa
Pig	ID Pig
Anomalia	ID Anomalia
Sezione	ID Sezione
Fabbrica	Partita IVA
Ditta Pig	Partita IVA
Interruzione	Corsa del Pig (ESTERNO)

Schema E-R finale



Schema logico

Pig(IDPig*, lunghezza, diametro, tipo, prezzo, numeroCorse, dataAcquisto, dittaPig[DittaPig])
CorsaPig(IDCorsa*, dataPartenza, dataArrivo, oraPartenza, oraArrivo, Pig[Pig])
Interruzione(CorsaPig[CorsaPig]*, kilometro, descrizione, dataOra)
DittaPig(PartitaIva*, ragioneSociale, formaGiuridica, telefono, sitoWeb, indirizzo, comune, provincia)
Anomalia(IDAnomalia*, progressiva, gravità, temperatura, dataManutenzione, scansione, descrizione, CorsaPig[CorsaPig], Sezione)
Sezione(IDSezione*, materiale, dataManutenzione, dataInstallazione, lunghezza, diametro, fabbrica)
FabbricaSezione(PartitaIva*, ragioneSociale, formaGiuridica, telefono, sitoWeb, indirizzo, comune, provincia)



Interazioni con il DB e programmabilità

Operazioni sul Database (Query)

Visualizzare il Pig che ha subito più interruzioni

```
SELECT COUNT(IDPig) as Interruzioni, IDPig
FROM Pig INNER JOIN CorsaPig CP 1<->1..n: on Pig.IDPig = CP.Pig
INNER JOIN Interruzione I 1<->1: on CP.IDCorsa = I.CorsaPig
GROUP BY IDPig ORDER BY Interruzioni DESC LIMIT 1;
```

Data una Ditta fornitrice di un Pig, ricavare il numero di volte che ha fornito Pig esterni

```
SELECT COUNT(PartitaIVA)
FROM DittaPig
INNER JOIN Pig P 1<->1..n: on DittaPig.PartitaIVA = P.DittaPig
WHERE DittaPig = ?;
```

Dato un Pig ricavare il numero di corse

```
SELECT COUNT(Pig) AS NumeroCorse
FROM CorsaPig
WHERE Pig = ?
GROUP BY Pig;
```

Ricavare le anomalie non ancora sottoposte a manutenzione

```
SELECT * FROM Anomalia
WHERE DataManutenzione IS NULL;
```

Data un'Anomalia, ricavare tutti i dati della sezione di tubo

```
SELECT s.* FROM Sezione s
INNER JOIN Anomalia A 1<->1..n: on s.IDSezione = A.Sezione
WHERE IDAnomalia = ?;
```

Codice Java

Visualizzare il Pig che ha subito più interruzioni

```
public String[] maxNumInterruzioni() throws SQLException {
    Statement statement = conn.createStatement();
    ResultSet result = statement.executeQuery( sql: "SELECT COUNT(IDPig) as Interruzioni, IDPig\n" +
            "    FROM Pig INNER JOIN CorsaPig CP on Pig.IDPig = CP.Pig\n" +
            "    INNER JOIN Interruzione I on CP.IDCorsa = I.CorsaPig\n" +
            "    GROUP BY IDPig ORDER BY Interruzioni DESC LIMIT 1;");

    result.next();
    int interruzioni = result.getInt( columnLabel: "Interruzioni");
    int idPig = result.getInt( columnLabel: "IDPig");

    return new String[] {Integer.toString(interruzioni), Integer.toString(idPig)};
}
```

Data una Ditta fornitrice di un Pig, ricavare il numero di volte che ha fornito Pig esterni

```
public String numPigEsterni(String partitaIva) throws SQLException {
    PreparedStatement preparedStatement = conn.prepareStatement( sql: "SELECT COUNT(PartitaIVA) AS num\n" +
            "FROM DittaPig\n" +
            "INNER JOIN Pig P on DittaPig.PartitaIVA = P.DittaPig\n" +
            "WHERE DittaPig = ?;");
    preparedStatement.setString( parameterIndex: 1, partitaIva);
    ResultSet result = preparedStatement.executeQuery();

    result.next();
    return result.getString( columnLabel: "num");
}
```

Ricavare le anomalie non ancora sottoposte a manutenzione

```
public List<Anomalia> manutenzioniAnomalie() throws SQLException {
    List<Anomalia> anomalie = new ArrayList<>();
    Statement statement = conn.createStatement();
    ResultSet resultSet = statement.executeQuery( sql: "SELECT *\n" +
            "FROM anomalia\n" +
            "WHERE DataManutenzione IS NULL;");

    int i = 0;
    while (resultSet.next()) {
        anomalie.set(i++, new Anomalia(resultSet.getInt( columnIndex: 1),
                resultSet.getFloat( columnIndex: 2), resultSet.getInt( columnIndex: 3),
                resultSet.getDate( columnIndex: 4), resultSet.getString( columnIndex: 5),
                resultSet.getString( columnIndex: 6), resultSet.getInt( columnIndex: 7),
                resultSet.getInt( columnIndex: 8)));
    }

    return anomalie;
}
```

Data un'Anomalia, ricavare tutti i dati della sezione di tubo

```
public Sezione datiSezioneAnomalia(int idAnomalia) throws SQLException {
    PreparedStatement preparedStatement = conn.prepareStatement( sql: "SELECT s.* FROM Sezione s\n" +
            "INNER JOIN Anomalia A on s.IDSezione = A.Sezione\n" +
            " WHERE IDAnomalia = ?;");
    preparedStatement.setInt( parameterIndex: 1, idAnomalia);
    ResultSet resultSet = preparedStatement.executeQuery();

    resultSet.next();
    return new Sezione(resultSet.getInt( columnIndex: 1), resultSet.getString( columnIndex: 2),
            resultSet.getDate( columnIndex: 3), resultSet.getDate( columnIndex: 4),
            resultSet.getFloat( columnIndex: 5), resultSet.getInt( columnIndex: 6),
            resultSet.getString( columnIndex: 7));
}
```

Vincoli e Triggers

I vincoli su Temperatura e Gravità della tabella Anomalia sono stati imposti direttamente durante la creazione della stessa.

```
constraint Gravita_CK
    check (`Gravita` between 1 and 10),
constraint Temperatura_CK
    check (`Temperatura` between -(20) and 150)
```

Distanza temporale tra due corse del Pig superiore a una settimana

```
CREATE TRIGGER DateCheck
    BEFORE INSERT ON corsapig
    FOR EACH ROW
        BEGIN
            IF NEW.DataPartenza <= DATE_ADD((SELECT MAX(DataPartenza) FROM corsapig), INTERVAL 7 DAY ) THEN
                SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT='DataPartenza deve distare più di una settimana dall ultima';
            END IF;
        END
```

Si è supposto che le corse del Pig siano inserite in ordine cronologico.

Consentire solo a Smart Pig (tipo=2) di rilevare Anomalie

```
CREATE TRIGGER SmartCheck
    BEFORE INSERT ON Anomalia
    FOR EACH ROW
    BEGIN
        DECLARE tipo INT;
        SET tipo = (SELECT P.tipo FROM Pig P
                INNER JOIN CorsaPig CP  1<->1..n: on P.IDPig = CP.Pig
                WHERE CP.IDCorsa = NEW.CorsaPig);
        IF tipo <> 2 THEN
                SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT='La corsa del Pig deve essere di uno Smart Pig';
        END IF;
    END;
```

Diametro del Pig inferiore al diametro della sezione

```
CREATE TRIGGER SezioneCK
    BEFORE INSERT ON anomalia
    FOR EACH ROW
    BEGIN
        DECLARE DiamPig INT;
        DECLARE DiamSez INT;
        SET DiamPig = (SELECT P.Diametro FROM Pig P
            INNER JOIN CorsaPig CP 1<->1..n: on P.IDPig = CP.Pig
            WHERE NEW.CorsaPig=CP.IDCorsa);
        SET DiamSez = (SELECT S.Diametro FROM Sezione S
            WHERE NEW.Sezione=S.IDSezione);
        IF DiamPig > DiamSez THEN
                SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT='Il diametro del Pig usato in
                                    questa corsa è maggiore del diametro della sezione';
            END IF;
        END;
```

Modifica dell'attributo numeroCorse [Pig]

```
CREATE TRIGGER AddCorsa
    AFTER INSERT ON CorsaPig
    FOR EACH ROW
    BEGIN
        UPDATE Pig SET NumeroCorse = NumeroCorse+1 WHERE IDPig = NEW.Pig;
    END;

CREATE TRIGGER SubCorsa
    AFTER DELETE ON CorsaPig
    FOR EACH ROW
    BEGIN
        UPDATE Pig SET NumeroCorse = NumeroCorse-1 WHERE IDPig = OLD.Pig;
    END;
```

DataManutenzione [Anomalia] deve essere successiva alla data di arrivo del Pig e successiva all'installazione della sezione

```
CREATE TRIGGER ManutenzioneCheck
    BEFORE INSERT ON Anomalia
    FOR EACH ROW
    BEGIN
        DECLARE DM DATE;
        DECLARE DI DATE;
        SET DM = (SELECT DataArrivo FROM CorsaPig WHERE IDCorsa = NEW.CorsaPig);
        SET DI = (SELECT DataInstallazione FROM sezione WHERE IDSezione = NEW.Sezione);
        IF NEW.DataManutenzione < DM THEN
                    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT='La data di manutenzione deve essere
                                                            successiva all arrivo del pig';
        END IF;
        IF NEW.DataManutenzione < DI THEN
                    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT='La data di manutenzione deve essere
                                                            successiva all installazione della sezione';
        END IF;
    END;
```

Update DataManutenzione [Sezione] dopo manutenzione dell'anomalia e reset a *null* dopo cancellazione (nel caso sia inserita per errore)

```
CREATE TRIGGER AddAnomalia
    AFTER INSERT ON Anomalia
    FOR EACH ROW
    BEGIN
        DECLARE DM DATE;
        SET DM = (SELECT DataManutenzione FROM Sezione WHERE IDSezione = NEW.Sezione);
        IF NEW.DataManutenzione > DM THEN
            UPDATE Sezione S SET S.DataManutenzione = NEW.DataManutenzione WHERE S.IDSezione = NEW.Sezione;
        END IF;
    END;

CREATE TRIGGER DelAnomalia
    AFTER DELETE ON Anomalia
    FOR EACH ROW
    BEGIN
        UPDATE Sezione S SET S.DataManutenzione = null WHERE S.IDSezione = OLD.Sezione;
    END;
```

È stato scelto di non inserire transazioni perché non c'erano operazioni che creassero conflitti e incongruenze nel database, e le operazioni che modificano i dati nel DB non sono così frequenti da poter interferire tra loro.

Non sono state utilizzate stored procedures perché le operazioni di lettura del database sono poco frequenti per richiedere un'alta velocità di esecuzione. Inoltre le operazioni necessarie sono state salvate come procedure Java (più facile per il debug).