Analisi della correlazione tra incidenti e tempo atmosferico a New York

Periodo temporale: 2015-2017

Danilo Deponti - 737838 Mattia Curatitoli - 735722

Introduzione

introduzione

Analisi dei dati - Incidenti

Sorgente dati: Incidenti a New York

Il dataset contenente gli incidenti stradali avvenuti a New York nel periodo temporale da noi analizzato è stato ottenuto attraverso la piattaforma *NYC Open Data* nella sezione *Sicurezza pubblica*. Il sito mette a disposizione numerosi sistemi per filtrare i dati sugli incidenti stradali, come quello per lasso temporale, utilizzato nel progetto. E' possibile esportare questi dati in formato *CSV, TSV, XML, RSS e RDF*

Informazioni raccolte

Analizzando il massiccio dataset ricavato, sono state rilevate le seguenti informazioni che lo caratterizzano:

Incidente

L'incidente è l' entità caratterizzante del dataset, questa entità fornisce informazioni come la chiave univoca dell' incidente, data, ora, posizione (latitudine, longitudine) e numero di morti/feriti coinvolti. Sia per i morti che per i feriti è possibile sapere se erano pedoni, ciclisti o conducenti. Inoltre, vi sono anche informazioni riguardanti il tipo di veicolo e in quale modo il relativo veicolo può avere provocato l' incidente.

Anagarafica Tipologia veicolo

Analizzando attentamente il dataset è stato riscontrato che la descrizione della tipologia del veicolo è identica e si ripete per moltissimi incidenti. Su circa 700000 incidenti analizzati sono state rilevate solamente 200 tipologie di veicolo distinte, tra le quali più della metà sono comprensibili. Molto probabilmente, per la maggior parte degli incidenti è stato compilato un form di reportistica dove la tipologia di veicolo era definita tra un elenco.

Anagrafica Scatenante

Come già illustrato nel precedente paragrafo, anche per lo scatenante è possibile distinguere un numero finito e molto ridotto (circa 60) di dati che caratterizzano quasi costantemente gli incidenti. Addirittura, per questa informazione, il numero di scatenanti rilevate che non hanno significato è ridotto allo 0.

Schema logico relazionale

I dati recuperati sono stati forniti in un unico e decisamente ricco csv. I dati, per ogni incidente, erano riportati su una sola riga e non vi era alcuna logica di riutilizzo dei dati. Inoltre, appunto per come sono stati archiviati i dati, non vi era alcuna logica di normalizzazione. E' stato quindi deciso di scomporre i dati ricavati nelle seguenti entità:

incidente (**id**, unique_key,numero_morti, numero_feriti, pedoni_morti, pedoni_feriti, ciclisti_morti, ciclisti_feriti, conducenti_morti, conducenti_feriti, *id_luogo*, data_incidente, ora_incidente, created_at, updated_at)
La relazione *incidenti* contiene tutte le informazioni che contraddistinguono l'incidente: data, ora, numero feriti, numero morti ed una distinzione di essi in pedoni, ciclisti e conducenti.

scatenante (**id**, descrizione)
Questa relazione contiene tutti i possibili motivi che hanno provocato l'incidente. Questa relazione, per mezzo del suo identificativo univoco intero, permette di associare il motivo scatenante all' incidente per mezzo di un' apposita tabella di collegamento.

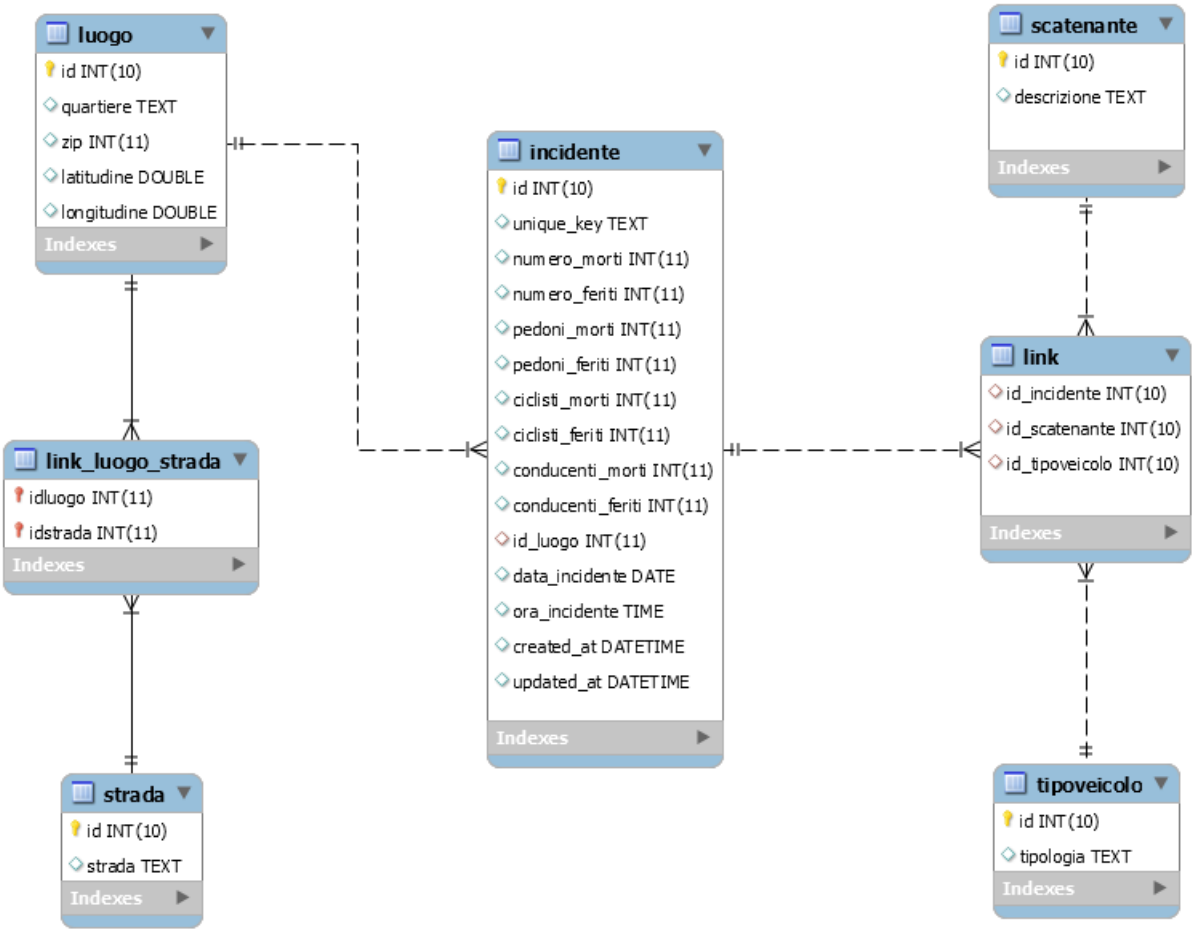
tipoVeicolo (**id**, tipologia)
La relazione *tipoVeicolo* definisce tutti i tipi di veicolo che possono caratterizzare l' incidente. Si collegano all' incidente, attraverso un identificativo univoco, per mezzo di un' apposita tabella di collegamento.

dettagli (*id_incidente*,*id_tipoveicolo*,*id_scatenante*)
Questa relazione è ciò che collega l' incidente con i veicoli ed, eventualmente, in che modo il suddetto veicolo ha provocato l'incidente. Il collegamento è creato tramite la tripletta di attributi $id_incidente$,$id_tipoveicolo$,$id_scatenante$. Ogni incidente può avere molteplici righe di dettaglio.

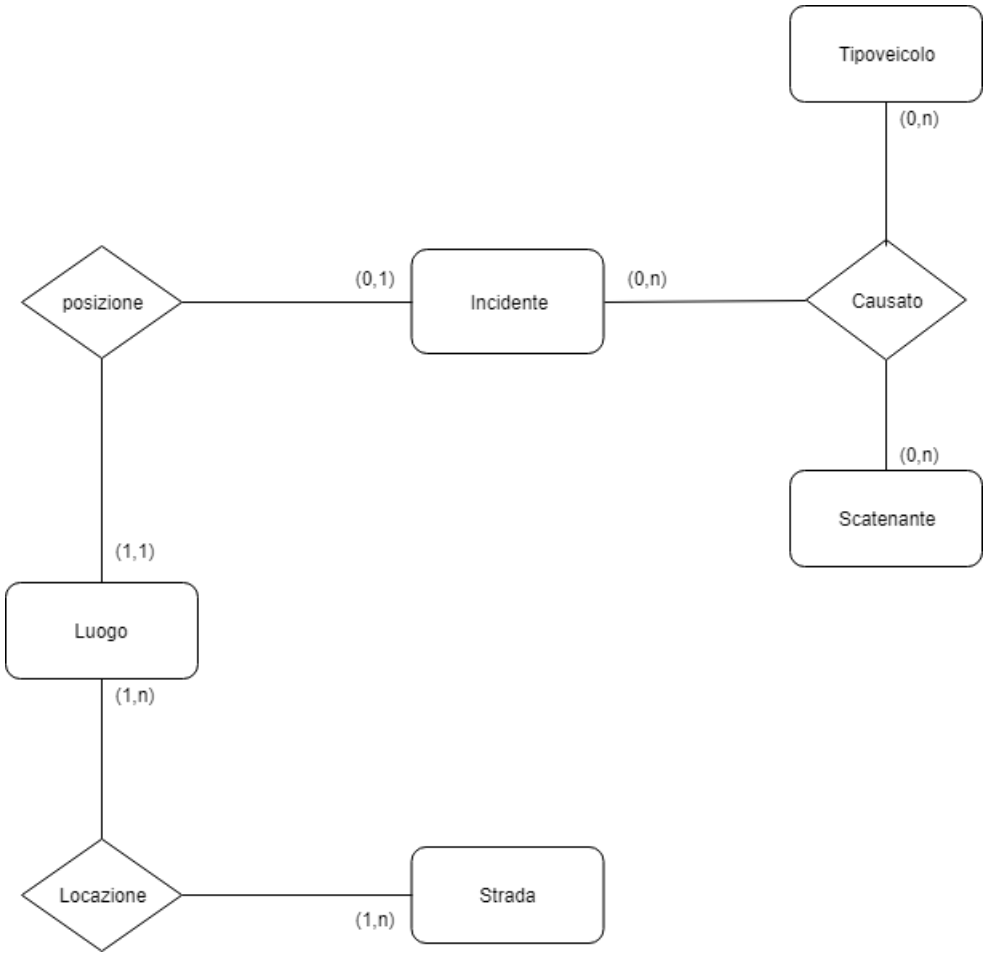
luogo (**id**, quartiere, zip, latitudine,longitudine)
La relazione luogo descrive la posizione in cui avviene l' incidente. Sebbene sia stata creata una chiave primaria intera incrementale (per facilitare il collegamento con la chiave esterna nella relazione *incidenti*) è stato applicato un vincolo di unicità sulla coppia *latitudine*,*longitudine*. E' possibile inoltre avere l'informazione del quartiere e dello zip (non sempre presenti)

strada (**id**, strada)
Un elenco di tutte le possibili strade presente a $New\ York$, ricopre il semplice ruolo di anagrafica.

link_luogo_strada (***id_luogo***,***id_strada***)
Questa relazione è il collegamento tra la posizione e la strada. Oltre ad essere chiavi esterne, la coppia di attributi che caratterizza la relazione funge a che da chiave univoca.


luogo
id INT(10)
quartiere TEXT
zip INT(11)
latitudine DOUBLE
longitudine DOUBLE
Indexes
link_luogo_strada
idluogo INT(11)
idstrada INT(11)
Indexes
strada
id INT(10)
strada TEXT
Indexes
incidente
id INT(10)
unique_key TEXT
numero_morti INT(11)
numero_feriti INT(11)
pedoni_morti INT(11)
pedoni_feriti INT(11)
ciclisti_morti INT(11)
ciclisti_feriti INT(11)
conducenti_morti INT(11)
conducenti_feriti INT(11)
id_luogo INT(11)
data_incidente DATE
ora_incidente TIME
created_at DATETIME
updated_at DATETIME
Indexes
scatenante
id INT(10)
descrizione TEXT
Indexes
link
id_incidente INT(10)
id_scatenante INT(10)
id_tipoveicolo INT(10)
Indexes
tipoveicolo
id INT(10)
tipologia TEXT
Indexes

Progettazione Concettuale



Importazione dei dati

Per l' importazione dei dati su database è stato utilizzato *Python*. La funzionalità di importazione si può dividere in 3 macro step:

- *Modellizzazione*: per rendere più semplice la comprensione delle tabelle che si andranno a creare su *DB*, sono state create diverse classi in *python* che sono strettamente legate ad esse. Vi sarà appunto la classe *Incidete*, *Luogo* ecc (fatta eccezione per le classi di collegamento come *dettagli* e $link_luogo_strada$)
- *Realizzazione del database*: creazione di un database *MySql* chiamato $db_incidenti$. Definizione delle relazioni chiamate *incidente*, *scatenante*, $tipoVeicolo$, *dettagli*, *luogo*, *strada* e $link_luogo_strada$ presenti nel database creato in precedenza, specificando gli attributi (nome e tipologia), le chiavi primarie, le chiavi esterne e la tipologia di motore *MySQL* (quella di default: *InnoDB*)

- *Implementazione metodo store*: ogni classe creata implementa un proprio metodo *store* che ne garantisce il salvataggio a *db*. Questo è stato fatto per rendere bene l' idea del collegamento *istanza - riga della tabella*

Analisi dei dati - Meteo

Sorgente dati: API Meteo a New York

I dato meteo sono stati reperiti da https://darksky.net. Questo sito mette a disposizione delle *API* che consentono di ottenere svariate informazioni riguardanti il meteo nella zona indicata. la *url* da chiamare per interrogare il servizio rest esposto è:

https://api.darksky.net/forecast/*token*/*lat*,*lon*,*datetime*

Dove:

- *token* è l' identificativo utente per poter usare le *API*
- *lat* e *lon* sono le coordinate della posizione per la quale si vogliono avere informazioni
- *datetime* è la data richiesta, scritta in formato *Y*-*m*-*d*T*h*:*i*:*s*

New York si trova alle coordinate 40 −74 (l' API riconosce la città dalla posizione datale). L' *API* fornisce informazioni sul meteo per tutte le 24 ore successive alla data indicata, con cadenza oraria. E' bastato quindi inserire la mezzanotte di ogni giorno per avere informazioni su tutte le ore della giornata.

Informazioni raccolte

Analizzando le informazioni sul meteo, è stato possibile ricavare la seguente informazione:

Meteo

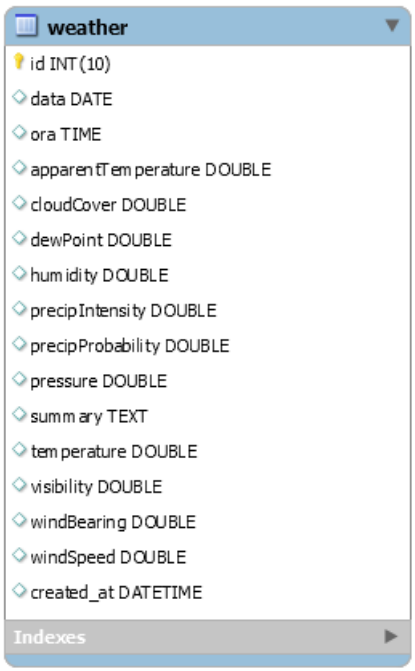
Questa entità è l' unica informazione ricavata. Essa contiene posizione, umidità, temperatura e molte altre informazioni riguardanti il meteo. Inoltre, per ogni orario, vi è una informazione riassuntiva sul meteo (*Rainy*, *Sunny*..) Nello specifico, abbiamo:

- *UNIX timestamp* della rilevazione meteo
- *temperatura*, *temperatura percepita* e *punto di rugiada* in gradi Fahrenheit
- *umidita'* (valore tra 0 ed 1) e *pressione* (millibars)
- *visibilita'* (miglia) e *copertura nuvole* (percentuale)
- *velocita'* (miglia orarie) ed *inclinazione* (in gradi, partendo da Nord in senso orario) del vento
- *riassunto* del meteo in quella data e quella ora

Schema logico relazionale

Essendo l' informazione fornita dalle *API* decisamente ben delineata ed "atomica". E' stato possibile delineare immediatamente l' entità *weather*.

weather (**id**, data,ora,apparentTemperature,cloudCover,dewPoint,humidity,precipIntensity,precipProb
visibility,windBearing,windSpeed,created_at)
Un elenco di tutte le possibili strade presente a *New York*, ricopre il semplice ruolo di anagrafica.



Progettazione Concettuale

Importazione dei dati

Come per la precedente sorgente dati, anche per il meteo è stato utilizzato uno script *Python* per popolare il database, nello specifico:

- *Modellizzazione*: mapping classe-tabella per ogni singola entità del database
- *Realizzazione del database*: creazione di un database *MySql* chiamato *meteo*. Definizione della relazione chiamata *weather* specificando gli attributi (nome e

tipologia), le chiavi primarie e la tipologia di motore *MySQL* (quella di default: *InnoDB*)

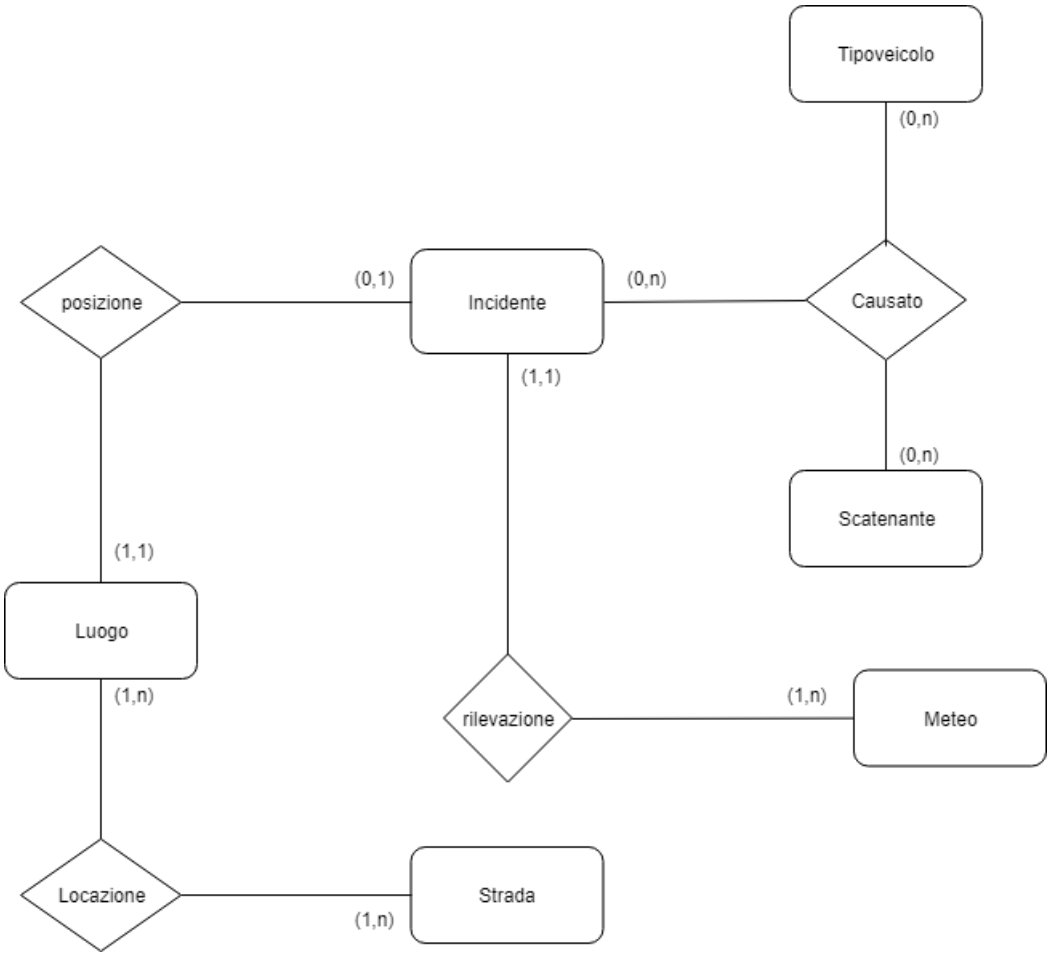
- *Implementazione metodo store*

Schema Riconciliato - ODS

In questo capitolo verrà illustrata l' analisi delle 2 fonti per poter infine produrre lo schema riconciliato ODS. In primo luogo verrà illustrato lo schema concettuale per poi ricavare lo schema logico-relazionale. Infine, verranno illustrate le metodologie di ETL ($Extract$, $Transform$, $Load$) per l'importazione dei dati dalle 2 sorgenti all' interno dell ODS.

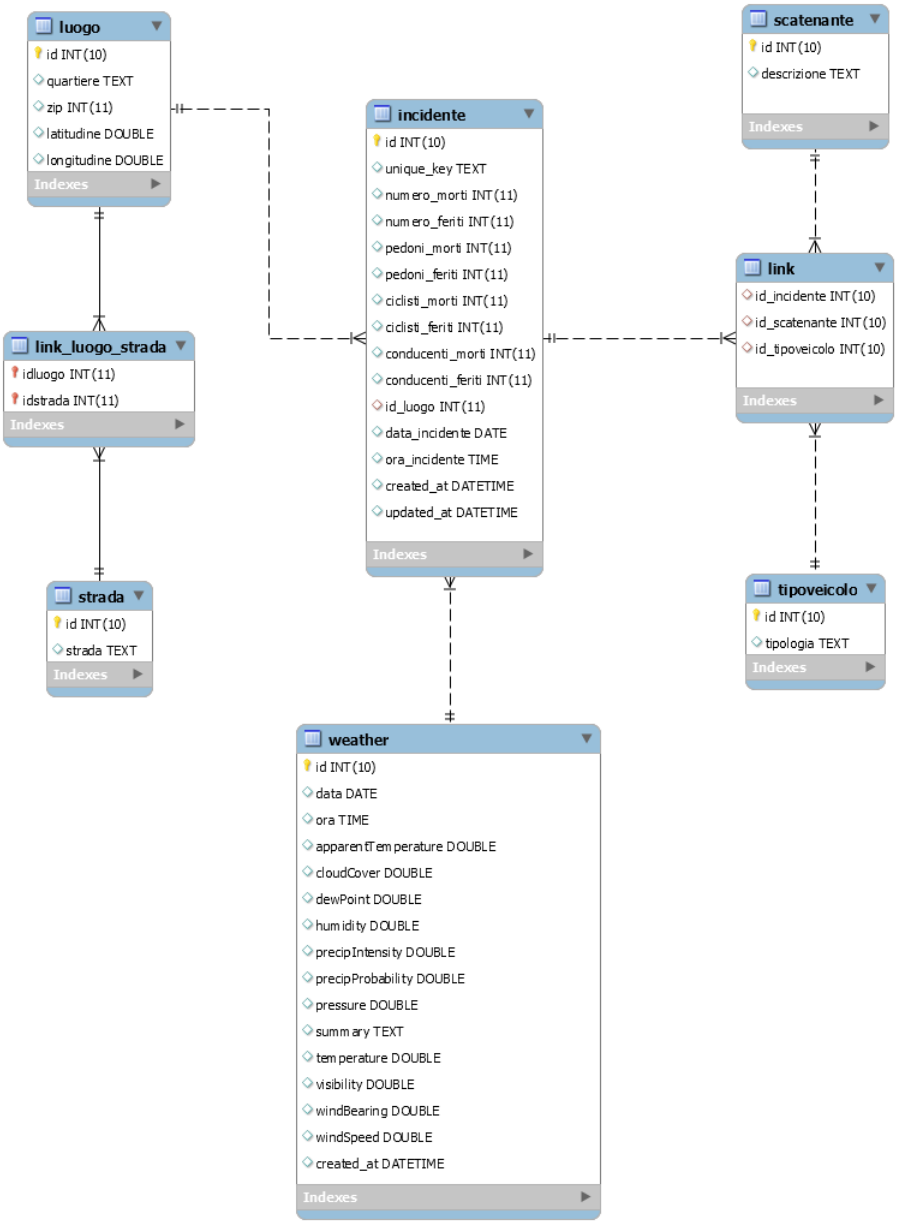
Progettazione Concettuale

A partire dagli schemi concettuali delle 2 sorgenti, è possibile ricreare lo schema concettuale riconciliato. In questo caso, il processo è di facile soluzione in quanto si sta trattando di sorgenti dati totalmente diverse tra loro e gli schemi delle sorgenti non si sovrappongono in alcun modo. Per poter effettuare questa interazione, è stato deciso di effettuare un collegamento tra l' entità *Incidente* e l' entità *Meteo*. Questa relazione è una relazione *uno a molti* (un incidente ha una sola rilevazione meteo, mentre possono esistere più incidenti per la stessa rilevazione meteo) e vede un collegamento tra la coppia di attributi (*weather.data*,*weather.ora*) e la coppia (*incidente.data*$_i$*ncidente*,*incidente.ora*$_i$*ncidente*)



Schema Logico - Relazionale

Lo schema logico-relazionale dell' ODS è stato realizzato a partire dallo schema concettuale dell' *ODS* e non è altro che una semplice unione degli schemi precedentemente realizzati.



Per creare l' *ODS* è stato utilizzato uno script *Python* e la sua relativa libreria *mySql* per la gestione dei *DB*.

Costruzione di un Data Warehouse

Questo capitolo tratterà l' effettiva creazione del *data warehouse*. Verrà illustrata la fase concettuale e quella logica con relativa scelta sello schema da seguire. Infine, verrà presentato il processo di trasformazione dei dati dall' *ODS* al *data warehouse*.

Progettazione Concettuale