Cofinanziato dall'Unione europea

REGIONE PIEMONTE

FinTech Software Developer- 2023/2025

Basi di dati SQL

Docente: Roi Davide Simone

Titolo argomento: Architettura Postgresql – Creazione di Tabelle e Viste

Roi Davide
Dispense

Intervento realizzato da
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Tipologie di Tabelle
(Tabelle dei fatti / dati)

Roi Davide Dispense

Questo tipo di tabella è funzionale a contenere i dati principali che vengono aggiornati regolarmente dall'utilizzo applicativo (es: ft_operazione, ft_acquisto, ft_passaggio, ft_cliente, ft_ordini, ft_pagamento, ecc..)

```
CREATE TABLE ft_cliente (
   nome VARCHAR(50), -- conterrà la descrizione dell'entità
   cognome VARCHAR(500), -- conterrà la descrizione dell'entità
   codice_fiscale VARCHAR(500), -- conterrà la descrizione dell'entità
   ecc..
   CONSTRAINT const_ft_cliente_pk PRIMARY KEY (cognome, codice_fiscale)
);

CREATE TABLE ft_cliente (
   codice SERIAL, -- conterrà il codice univoco dell'entità
   nome VARCHAR(50), -- conterrà la descrizione dell'entità
   cognome VARCHAR(500), -- conterrà la descrizione dell'entità
   codice_fiscale VARCHAR(500), -- conterrà la descrizione dell'entità
   ecc..
   CONSTRAINT const_ft_cliente_pk PRIMARY KEY (codice),
   CONSTRAINT const_ft_cliente_unique UNIQUE (cognome, codice_fiscale)
);
```

IMPORTANTE: Valutare attentamente se, in una tabella principale dei dati, sia meglio definire come PRIMARY KEY le colonne che contengono i «dati reali» che identificano univocamente il record (es: cognome e codice_fiscale) oppure se INVENTARE un progressivo sequenziale (SERIAL) e di conseguenza utilizzare il vincolo di UNIQUE per evitare che le colonne dei dati reali possano essere duplicate creando una incoerenza nei contenuti.
(es: se nel secondo esempio non creiamo la UNIQUE, rischiamo che venga inserito più volte lo stesso cliente, soltanto con codice diverso)

Nella pratica comune spesso si predilige l'utilizzo delle colonne dei dati reali come primary key, ma a volte per avere prestazioni migliori di lettura dei dati viene adottata la soluzione del progressivo sequenziale in quando è un INTEGER quindi più performante rispetto ad una lista di campi VARCHAR.

Intervento realizzato da ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Tipologie di Tabelle

(Tabelle di dimensione / decodifica / configurazione)

Roi Davide Dispense

Questo tipo di tabella è funzionale a gestire liste di valori con le relative decodifiche (es: dm_codifica_regione, dm_tipologia_pagamento, dm_anagrafica_operazioni, dm_configurazione_postazioni, ecc..). Sono tabelle che tendenzialmente una volta configurate non crescono di dimensione se non con aggiornamenti sporadici.

```
CREATE TABLE dm_nome_tabella (
   codice SERIAL, -- conterrà il codice univoco dell'entità
   descrizione VARCHAR(500), -- conterrà la descrizione dell'entità
   attributo_1 [ BOOLEAN, FLOAT, VARCHAR, INTEGER, ecc ], -- conterrà ulteriori attributi del record
   ecc..
   CONSTRAINT const_dm_nome_tabella_pk PRIMARY KEY (codice),
   CONSTRAINT const_dm_nome_tabella_unique UNIQUE (descrizione)
);

CREATE TABLE dm_nome_tabella (
   codice [ INTEGER | VARCHAR ], -- conterrà il codice univoco dell'entità
   descrizione VARCHAR(500), -- conterrà la descrizione dell'entità
   attributo_1 [ BOOLEAN, FLOAT, VARCHAR, INTEGER, ecc ], -- conterrà ulteriori attributi del record
   ecc..
  CONSTRAINT const_dm_nome_tabella_pk PRIMARY KEY (codice),
  CONSTRAINT unique_dm_nome_tabella UNIQUE (descrizione)
);
```

In questo tipo di soluzione il campo «codice» viene utilizzato dalle altre tabelle come FOREIGN KEY per collegare i record codificati.
Il vincolo di UNIQUE viene utilizzato per evitare che vengano duplicati su più record dei valori che fondamentalmente dovrebbero essere univoci (in questo caso impediamo che si presentino 2 descrizioni identiche per due codici diversi.

Tipologie di Tabelle
(Tabelle di Stack)

Roi Davide Dispense

Questo tipo di tabella è funzionale quando si ha necessità di creare delle tabelle temporanee per spostare dati da un sistema ad un altro. Dette anche tabelle FIFO (First Input, First Output) permettono di garantire che l'ordine di estrazione dei dati segua temporalmente lo stesso ordine con il quale i dati sono stati inseriti in modo che i dati più vecchi vengano estratti per primi.

```
CREATE TABLE ft_nome_tabella_stack (
   id_stack SERIAL, -- definisce la colonna con valore univoco auto-incrementale per ogni riga
   campo_1 VARCHAR(50), -- definisce una colonna "nome" di tipo stringa
   campo_2 VARCHAR(50), -- definisce una colonna "cognome" di tipo stringa
   ecc…
);
```

Il codice della procedura che si occuperà di estrarre i dati dovrà:
Step1: Leggere i valori **minimo** e **massimo** della colonna id_stack presenti nella tabella nome_tabella_stack e registrarli in due variabili.
Step2: Estrarre i record ordinandoli dal valore id_stack più piccolo fino al valore massimo letto.
Step3: Se tutto è andato bene, cancellare i record della tabella nome_tabella_stack che vanno dal valore minimo al valore massimo estratti.

Fonti:

Roi Davide Dispense

- Documentazione ufficiale di Postgresql
  https://www.postgresql.org/docs/
- SQL online tutorial.org
  https://www.sqltutorial.org/
- SQL online documentazione w3schools
  https://www.w3schools.com/
- Autore: Serena Sensini (2021)
  Titolo: Dasi di Dati - Tecnologie, architetture e linguaggi per database
  Editore: Apogeo
  ISBN: 9788850335534

Intervento realizzato da ITS Tecnologie dell'Informazione e della Comunicazione

Fine della Presentazione

Intervento realizzato da
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE