

# A Primer for the pfx\_dw Data Warehouse Schema

The **pfx\_dw** schema is a fairly textbook [star schema](#), with things like jobs and worker usage in **fact** tables, and the various things that you might use in a SQL **WHERE** clause in **dimension** tables.

## The job\_fact table, a typical "fact" table.

For example, to get reports about various jobs over time, you'll be querying the **pfx\_dw.job\_fact** table:

```
mysql> SHOW CREATE TABLE pfx_dw.job_fact\G
***** 1. row *****
Table: job_fact
Create Table: CREATE TABLE `job_fact` (
  `time_sk` int(11) NOT NULL,
  `job_id` int(11) DEFAULT NULL,
  `job_name` varchar(255) DEFAULT NULL,
  `job_priority` int(11) DEFAULT NULL,
  `job_timesubmit` int(11) DEFAULT NULL,
  `job_timestart` int(11) DEFAULT NULL,
  `job_timecomplete` int(11) DEFAULT NULL,
  `jobstatus_sk` int(11) DEFAULT NULL,
  `user_sk` smallint(5) unsigned DEFAULT NULL,
  `cluster_sk` smallint(5) unsigned DEFAULT NULL,
  `account_sk` smallint(5) unsigned DEFAULT NULL,
  `kind_sk` smallint(5) unsigned DEFAULT NULL,
  `prototype_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_show_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_shot_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_seq_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_client_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_dept_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_custom1_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_custom2_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_custom3_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_custom4_sk` smallint(5) unsigned DEFAULT NULL,
  `prod_custom5_sk` smallint(5) unsigned DEFAULT NULL,
  `cpu_count` int(10) unsigned NOT NULL,
  `cpu_seconds` int(10) unsigned NOT NULL,
  `work_count` int(10) unsigned NOT NULL,
  `worktime_min` int(10) unsigned NOT NULL,
  `worktime_max` int(10) unsigned NOT NULL,
  `worktime_avg` int(10) unsigned NOT NULL,
  `mem_max` bigint(20) unsigned DEFAULT NULL,
  `mem_avg` bigint(20) unsigned DEFAULT NULL,
  `mem_stddev` bigint(20) unsigned DEFAULT NULL,
  UNIQUE KEY `job_id` (`job_id`),
  KEY `time_sk` (`time_sk`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1
1 row in set (0.00 sec)
```



**The "job\_time\*" columns are stored in UNIX epoch time**

To convert them to human-readable format use the MySQL `FROM_UNIXTIME()` function.

```
mysql> SELECT job_id, job_timesubmit, FROM_UNIXTIME(job_timesubmit) FROM pfx_dw.job_fact WHERE job_id=
+-----+-----+-----+
| job_id | job_timesubmit | FROM_UNIXTIME(job_timesubmit) |
+-----+-----+-----+
| 98269 | 1414709214 | 2014-10-30 15:46:54 |
+-----+-----+-----+
```

## "\*\_sk" columns can be used to do INNER JOINs to a similarly named dimension table

Any column that is named with an **\_sk** suffix is a **S**ynthetic **K**ey that points to a corresponding dimension table, named with the part of the column before the **\_sk**; the dimension table will have a **\_dim** suffix in the name. This way, it's easy to write the **JOIN**'s, the column name is a clue to the dimension table, which will have a column of the same name. Almost every dimension table will consist of a **\*\_sk** **PRIMARY KEY** and a **name** column.

## A typical dimension table, the "user\_dim" table

For example, the **user\_sk** column can be used to do a SQL **INNER JOIN** to the **user\_dim** table.

```
mysql> SHOW CREATE TABLE pfx_dw.user_dim\G
***** 1. row *****
Table: user_dim
Create Table: CREATE TABLE `user_dim` (
  `user_sk` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  PRIMARY KEY (`user_sk`),
  UNIQUE KEY `name` (`name`)
) ENGINE=MyISAM AUTO_INCREMENT=10 DEFAULT CHARSET=latin1
1 row in set (0.00 sec)

mysql> SELECT name FROM user_dim;
+-----+
| name |
+-----+
| Administrator |
| bar |
| foo |
| foobar |
| fubar |
| jburk |
| perforce |
| root |
| shinya |
+-----+
```

Get a count of all jobs for a particular user:

```
mysql> SELECT
COUNT(*)
FROM
  job_fact AS fact
INNER JOIN
  user_dim AS user
ON
  fact.user_sk = user.user_sk
WHERE
  user.name = 'jburk'
;
```

COUNT(*)
5868

## The time dimension table

The **pfx\_dw.time\_dim** table is provided so that you don't have to perform date/time operations on every row in a **fact** table (since they can run into the 100's of millions of rows), instead you do a SQL `INNER JOIN` to it and use the values in the **time\_dim** table in your `WHERE` clause. The **time\_sk** column in every fact table has an identical value in the `time_dim` table which has a single row with a primary key `time_sk`. The `time_sk` value is actually the unix epoch time in seconds:

```
mysql> SELECT
*
FROM
  time_dim
ORDER BY
  time_sk
DESC LIMIT 1
;
+-----+-----+-----+-----+-----+-----+-----+-----+
---+
| time_sk | date_time          | hour | date       | dow | month_name | month | year |
+-----+-----+-----+-----+-----+-----+-----+-----+
---+
| 1392339600 | 2014-02-13 17:00:00 | 17    | 2014-02-13 | 5    | February   | 2      | 2014 |
+-----+-----+-----+-----+-----+-----+-----+-----+
---+
1 row in set (0.00 sec)

mysql> SELECT
  FROM_UNIXTIME(time_sk)
, date_time
FROM
  time_dim
ORDER BY
  time_sk
DESC LIMIT 1
;
+-----+-----+
| FROM_UNIXTIME(time_sk) | date_time          |
+-----+-----+
| 2014-02-13 17:00:00    | 2014-02-13 17:00:00 |
+-----+-----+
```

## The "job status" dimension table

The **pfx\_dw.jobstatus\_dim** table is one of the few exceptions to the normal dimension table structure; it provides a mapping between the integer and human-readable status values.

```
mysql> SELECT * FROM jobstatus_dim;
```

jobstatus_sk	status_int	status_char	effective_date	expiry_date
1	16	complete	1999-12-31	9999-12-31
2	32	failed	1999-12-31	9999-12-31
3	48	killed	1999-12-31	9999-12-31
4	272	blocked	1999-12-31	9999-12-31
5	288	waiting	1999-12-31	9999-12-31
6	304	suspended	1999-12-31	9999-12-31
7	320	pending	1999-12-31	9999-12-31
8	323	waiting	1999-12-31	9999-12-31
9	325	badlogin	1999-12-31	9999-12-31
10	336	running	1999-12-31	9999-12-31
11	261	dying	1999-12-31	9999-12-31

**Get a count of all jobs for a particular user for January, 2014:**

```
mysql> SELECT
COUNT(*)
FROM
  job_fact AS fact
INNER JOIN
  user_dim AS user
ON
  fact.user_sk = user.user_sk
INNER JOIN
  time_dim AS time
ON
  fact.time_sk=time.time_sk
WHERE
  user.name = 'jburk'
AND
  time.month = 1
AND
  time.year = 2014
;
```

COUNT(*)
83

**Get a count of all jobs for each user for all of 2013:**

```

mysql> SELECT
  user.name
  , time.month_name
  , COUNT(*) as "job count"
FROM
  job_fact AS fact
INNER JOIN
  user_dim AS user
ON
  fact.user_sk = user.user_sk
INNER JOIN
  time_dim AS time
ON
  fact.time_sk=time.time_sk
WHERE
  time.year = 2013
GROUP BY
  user.name
  , time.month
ORDER BY
  user.name
  , time.month
;

```

name	month_name	job count
Administrator	January	4
bar	July	1
foo	July	2
foobar	July	1
foobar	September	2
fubar	July	1
jburk	March	1
jburk	May	123
jburk	June	24
jburk	July	220
jburk	August	140
jburk	September	68
jburk	October	560
jburk	November	59
jburk	December	4467
perforce	January	128
perforce	February	4
root	January	31
root	February	37
root	March	7
root	April	17
root	May	7
root	June	1
root	July	11
root	September	2
root	December	2
shinya	January	23
shinya	February	69
shinya	March	7
shinya	April	20
shinya	May	2

**Get a count of all jobs for each user for all of 2013, broken down by month and the job's final status:**

```
mysql> SELECT
  user.name
, time.year
, time.month_name
, status.status_char
, COUNT(*) as "job count"
FROM
  job_fact AS fact
INNER JOIN
  user_dim AS user
ON
  fact.user_sk=user.user_sk
INNER JOIN
  time_dim AS time
ON
  fact.time_sk=time.time_sk
INNER JOIN
  jobstatus_dim AS status
ON
  fact.jobstatus_sk=status.jobstatus_sk
WHERE
  time.year = 2013
GROUP BY
  user.name
, time.month
, status.status_int
ORDER BY
  user.name
, time.month
,status.status_char
;
```

name	year	month_name	status_char	job count
Administrator	2013	January	failed	4
bar	2013	July	complete	1
foo	2013	July	complete	2
foobar	2013	July	complete	1
foobar	2013	September	complete	2
fubar	2013	July	complete	1
jburk	2013	March	complete	1
jburk	2013	May	complete	100
jburk	2013	May	failed	11
jburk	2013	May	killed	12
jburk	2013	June	complete	4
jburk	2013	June	failed	20
jburk	2013	July	complete	134
jburk	2013	July	failed	33
jburk	2013	July	killed	53
jburk	2013	August	complete	67
jburk	2013	August	failed	25
jburk	2013	August	killed	48
jburk	2013	September	complete	38
jburk	2013	September	failed	17
jburk	2013	September	killed	13

jburk	2013	October	complete	450
jburk	2013	October	failed	61
jburk	2013	October	killed	49
jburk	2013	November	complete	38
jburk	2013	November	failed	12
jburk	2013	November	killed	9
jburk	2013	December	complete	3812
jburk	2013	December	failed	627
jburk	2013	December	killed	28
perforce	2013	January	failed	46
perforce	2013	January	killed	82
perforce	2013	February	complete	3
perforce	2013	February	killed	1
root	2013	January	complete	24
root	2013	January	failed	6
root	2013	January	killed	1
root	2013	February	complete	34
root	2013	February	killed	3
root	2013	March	complete	7
root	2013	April	complete	9
root	2013	April	failed	4
root	2013	April	killed	4
root	2013	May	complete	6
root	2013	May	killed	1
root	2013	June	complete	1
root	2013	July	complete	7
root	2013	July	failed	1
root	2013	July	killed	3
root	2013	September	complete	1
root	2013	September	failed	1
root	2013	December	complete	2
shinya	2013	January	complete	4
shinya	2013	January	failed	7
shinya	2013	January	killed	12
shinya	2013	February	complete	60
shinya	2013	February	failed	2
shinya	2013	February	killed	7
shinya	2013	March	complete	2
shinya	2013	March	failed	2
shinya	2013	March	killed	3
shinya	2013	April	complete	16



	shinya		2013		April		killed		4	
	shinya		2013		May		killed		2	
+-----+-----+-----+-----+-----+										

Get the sum total of `cpu_seconds` used for each user for the last 7 days, broken down by user, date, and the job's final status:

```

SELECT
    user.name
      , time.date
      , status.status_char
      , SUM(fact.cpu_seconds) as "cpu_time"
FROM
    job_fact AS fact
INNER JOIN
    user_dim AS user
ON
    fact.user_sk=user.user_sk
INNER JOIN
    time_dim AS time
ON
    fact.time_sk=time.time_sk
INNER JOIN
    jobstatus_dim AS status
ON
    fact.jobstatus_sk=status.jobstatus_sk
WHERE
    DATEDIFF(CURDATE(), time.date_time) < 7
GROUP BY
    user.name
      , time.date
      , status.status_int
ORDER BY
    time.date
      , cpu_time DESC
      , status.status_char
;

```

```

+-----+-----+-----+-----+
| name   | date       | status_char | cpu_time |
+-----+-----+-----+-----+
<<  snipped  >>
| jburk  | 2014-07-14 | complete    | 351036   |
| jburk  | 2014-07-14 | killed      | 60029    |
| jburk  | 2014-07-14 | failed      | 139      |
| coxj   | 2014-07-14 | killed      | 98        |
| garza  | 2014-07-14 | killed      | 0         |
| jburk  | 2014-07-15 | complete    | 28910    |
| fubar  | 2014-07-15 | complete    | 18610    |
| foobar | 2014-07-15 | complete    | 18561    |
| jburk  | 2014-07-15 | killed      | 16967    |
| jburk  | 2014-07-15 | failed      | 27        |
| jburk  | 2014-07-16 | complete    | 46797    |
| jburk  | 2014-07-16 | killed      | 17136    |
| jburk  | 2014-07-16 | failed      | 2         |
<<  snipped  >>
+-----+-----+-----+-----+

```